

# Lecture Notes in Computer Science

1815

Guy Pujolle Harry Perros Serge Fdida  
Ulf Körner Ioannis Stavrakakis (Eds.)

## NETWORKING 2000

**Broadband Communications,  
High Performance Networking,  
and Performance of  
Communication Networks**

**IFIP-TC6/European Commission International Conference  
Paris, France, May 2000  
Proceedings**



IFIP TC6



Springer

# Lecture Notes in Computer Science

Edited by G. Goos, J. Hartmanis and J. van Leeuwen

1815

**Springer**

*Berlin*

*Heidelberg*

*New York*

*Barcelona*

*Hong Kong*

*London*

*Milan*

*Paris*

*Singapore*

*Tokyo*

Guy Pujolle Harry Perros Serge Fdida  
Ulf Körner Ioannis Stavrakakis (Eds.)

# NETWORKING 2000

## Broadband Communications, High Performance Networking, and Performance of Communication Networks

IFIP-TC6/European Commission International Conference  
Paris, France, May 14-19, 2000  
Proceedings



Springer



## Series Editors

Gerhard Goos, Karlsruhe University, Germany  
Juris Hartmanis, Cornell University, NY, USA  
Jan van Leeuwen, Utrecht University, The Netherlands

## Volume Editors

Guy Pujolle

University of Versailles, PRiSM Laboratory  
45, Avenue des Etats-Unis, 78035 Versailles, France  
Email: gp@prism.uvsq.fr

Harry Perros

North Carolina State University, College of Engineering  
Department of Computer Science, Raleigh, NC 27695-7534, USA  
Email: hp@unity.ncsu.edu

Serge Fdida

Université Pierre et Marie Curie, Laboratoire LIP6-CNRS  
8, rue du Capitaine Scott, 75015 Paris, France  
Email: serge.fdida@lip6.fr

Ulf Körner

Lund University, Department of Communication Systems  
Box 118, 221 00 Lund, Sweden  
Email: ulfk@tts.lu.se

Ioannis Stavrakakis

University of Athens, Department of Informatics  
Panepistimiopolis, Illisia, 157 84 Athens, Greece  
Email: istavrak@di.uoa.gr

## Cataloging-in-Publication Data applied for

Die Deutsche Bibliothek - CIP-Einheitsaufnahme

Networking 2000 : broadband communications, high performance

networking, and performance of communication networks ;

IFIP-TC6-European Union international conference, Paris, France, May

14 - 19, 2000 ; proceedings / Guy Pujolle ... (ed.). - Berlin ;

Heidelberg ; New York ; Barcelona ; Hong Kong ; London ; Milan ; Paris

; Singapore ; Tokyo : Springer, 2000

(Lecture notes in computer science ; 1815)

ISBN 3-540-67506-X

CR Subject Classification (1998): C.2, C.4, D.2, H.4.3, J.2, J.1, K.6

ISSN 0302-9743

ISBN 3-540-67506-X Springer-Verlag Berlin Heidelberg New York

This work is subject to copyright. All rights are reserved, whether the whole or part of the material is concerned, specifically the rights of translation, reprinting, re-use of illustrations, recitation, broadcasting, reproduction on microfilms or in any other way, and storage in data banks. Duplication of this publication or parts thereof is permitted only under the provisions of the German Copyright Law of September 9, 1965, in its current version, and permission for use must always be obtained from Springer-Verlag. Violations are liable for prosecution under the German Copyright Law.

Springer-Verlag is a company in the BertelsmannSpringer publishing group.

© Springer-Verlag Berlin Heidelberg 2000

Printed in Germany

Typesetting: Camera-ready by author, data conversion by Steingraeber Satztechnik GmbH, Heidelberg

Printed on acid-free paper SPIN: 10720301 06/3142 5 4 3 2 1 0

## Preface

This was the first conference jointly organized by the IFIP Working Groups 6.2, 6.3, and 6.4. Each of these three Working Groups has its own established series of conferences. Working Group 6.2 sponsors the *Broadband Communications* series of conferences (Paris 1995, Montreal 1996, Lisboa 1997, Stuttgart 1998, and Hong-Kong 1999). Working Group 6.3 sponsors the *Performance of Communication Systems* series of conferences (Paris 1981, Zürich 1984, Rio de Janeiro 1987, Barcelona 1990, Raleigh 1993, Istanbul 1995, and Lund 1998). Working Group 6.4 sponsors the *High Performance Networking* series of conferences (Aaren 1987, Liège 1988, Berlin 1990, Liège 1992, Grenoble 1994, Palma 1995, New York 1997, Vienna 1998). It is expected that this new joint conference will take place every two years.

In view of the three sponsoring Working Groups, there were three separate tracks, one per Working Group. Each track was handled by a different co-chairman. Specifically, the track of Working Group 6.2 was handled by Ulf Körner, the track of Working Group 6.3 was handled by Ioanis Stavrakakis, and the track of Working Group 6.4 was handled by Serge Fdida. The overall program committee chairman was Harry Perros, and the general conference chairman was Guy Pujolle.

A total of 209 papers were submitted to the conference of which 82 were accepted. Each paper was submitted to one of the three tracks. Papers within each track were reviewed separately, and the decision as to which papers to accept was taken separately within each track. For presentation purposes, most of the sessions in the final program were made up by mixing papers from the three tracks.

The program covers a variety of research topics which are of current interest, such as wireless networks, optical networks, switching architectures, residential access networks, signaling, voice and video modelling, congestion control, call admission control, QoS, TCP/IP over ATM, interworking of IP and ATM, internet protocols, differentiated services, routing, multicasting, real-time traffic management, resource management and allocation, and performance modelling.

## VI Preface

We would like to thank the program committee members and the referees. Without their dedication and hard work this conference would not have materialized. We are also indebted to our two web masters Costas Tsibanis and Dimitris Koutoulas (with special thanks to Costas) at the Network Operation Center of the University of Athens, for setting-up and managing the conference's web site.

March 2000

Guy Pujolle  
Harry Perros  
Serge Fdida  
Ulf Körner  
Ioannis Stavrakakis

# **Executive Committee**

## **General Chair**

*Guy Pujolle - France*

## **Steering Committee**

*André Danthine - Belgium*

## **General Manager**

*Eric Horlait - France*

## **Technical Program Chair**

*Harry Perros - USA*

## **Special Program Chair: Broadband Communications**

*Ulf Körner*

## **Special Program Chair: Performance of Communication Systems**

*Ioannis Stavrakakis - Greece*

## **Special Program Chair: High Performance Networking**

*Serge Fdida - France*

## **Publicity Program Chair**

*Raouf Boutaba - Canada*

## **Tutorial Chair**

*Eric Horlait - France*

## **Local Arrangement and Registration Chair**

*Catherine Plottier - France*

## **Organizing Chair**

*Jean-Alain Hernandez - France*

*René Joly - France*

## Program Committee Members

*Augusto Albuquerque - Brazil*  
*Harmen van As - Austria*  
*Augusto Casaca - Portugal*  
*Paolo Castelli - Italy*  
*Imrich Chlamtac - USA*  
*Jean-Pierre Coudreuse - France*  
*Nelson L. S. Fonseca - Brazil*  
*Luigi Fratta - Italy*  
*Giorgio Gallassi - Italy*  
*Andre Girard - Canada*  
*Villy B. Iversen - Denmark*  
*Bijan Jabbari - USA*  
*Konosuke Kawashima - Japan*  
*Peter Key - UK*  
*Daniel Kofman - France*  
*Paul Kuehn - Germany*  
*Helmut Leopold - Austria*  
*John Luetchford - Canada*  
*Lorne Mason - Canada*  
*Serafim Nunes - Portugal*  
*Guido Petit - Belgium*  
*Sathya Rao - Switzerland*  
*Joao Rodrigues - Portugal*  
*Catherine Rosenberg - UK*  
*Tadao Saito - Japan*  
*Amardeo Sarnea - Germany*  
*Marion Schreinemachers - Netherl.*  
*Jan Slavik - Czech Republic*  
*Samir Tohme - France*  
*Danny Tsang - Hong Kong*  
*Finn Arve Aagesen - Norway*  
*Andres Albanese - USA*  
*Chase Bailey - USA*  
*Ermanno Berruto - Italy*  
*Andrew Campbell - USA*  
*Jon Crowcroft - UK*  
*Andre Danthine - Belgium*  
*Walid Dabbous - France*  
*Michel Diaz - France*  
*Sarolta Dibuz - Hungary*  
*Christophe Diot - USA*

*Otto Duarte - Brazil*  
*Wolfgang Effelsberg - Germany*  
*Nicolas D. Georganas - Canada*  
*Enrico Gregori - Italy*  
*Roch Guerin - USA*  
*Christian Huitema - USA*  
*Marjory Johnson - USA*  
*Farouk Kamoun - Tunisia*  
*Koos Koen - RSA*  
*Jacques Labetoulle - France*  
*Jean-Yves Le Boudec - Switzerl.*  
*Guy Leduc - Belgium*  
*Olli Martikainen - Finland*  
*Steve Pink - Sweden*  
*Nina Taft Plotkin - USA*  
*Radu Popescu-Zeletin - Germany*  
*Luigi Rizzo - Italy*  
*Aruna Seneviratne - Australia*  
*Otto Spaniol - Germany*  
*Ralf Steinmetz - Germany*  
*David Hutchison - UK*  
*Chris Blondia - Belgium*  
*Miklos Boda - Hungary*  
*Herwig Bruneel - Belgium*  
*Olga Casals - Spain*  
*Prosper Chemouil - France*  
*Giovanni Colombo - Italy*  
*Tony Ephremides - USA*  
*Andras Farago - USA*  
*Erol Gelenbe - USA*  
*Mario Gerla - USA*  
*Fabrice Guillemin - France*  
*Gerard Hebuterne - France*  
*Demetres Kouvatsos - UK*  
*Karl Lindberger - Sweden*  
*Jon Mark - Canada*  
*Marco Marsan - Italy*  
*Lazaros Merakos - Greece*  
*Jouni Mikkonen - Finland*  
*Debasis Mitra - USA*  
*Naotama Morita - Japan*

*Arne Nilsson - USA*  
*Raif Onvural - USA*  
*Ramon Puigjaner - Spain*  
*James Roberts - France*  
*Yutaka Takahashi - Japan*

*Ahmed Tantawy - USA*  
*Phouk Tran-Gia - Germany*  
*Jorma Virtamo - Finland*  
*Adam Wolisz - Germany*  
*Lyman Chapin -USA*

## List of Reviewers

*Kostas Tsibanis*  
*Harmen van As*  
*Augusto Albuquerque*  
*Lazaros Merakos*  
*Augusto Casaca*  
*Imrich Chlamtac*  
*Nelson L.S.Fonseca*  
*Luigi Fratta*  
*Giorgio Gallassi*  
*Andre Girard*  
*Villy B.Iversen*  
*Bijan Jabbari*  
*Konosuke Kawashima*  
*Peter Key*  
*Daniel Kofman*  
*Paul Kuehn*  
*Helmut Leopold*  
*Lorne Mason*  
*Serafim Nunes*  
*Tadao Saito*  
*Danny Tsang*  
*Finn Arve Aagesen*  
*Andres Albanese*  
*Chase Bailey*  
*Ermanno Berruto*  
*Andrew Campbell*  
*Jon Crowcroft*  
*Andre Danthine*  
*Walid Dabbous*  
*Michel Diaz*  
*Sarolta Dibuz*  
*Christophe Diot*  
*Otto Duarte*  
*Wolfgang Effelsberg*  
*Nicolas D. Georganas*  
*Enrico Gregori*  
*Roch Guerin*  
*Christian Huitema*  
*Marjory Johnson*  
*Farouk Kamoun*

*Koos Koen*  
*Jacques Labetoulle*  
*Jean-Yves Le Boudec*  
*Guy Leduc*  
*Olli Martikainen*  
*Steve Pink*  
*Nina Taft Plotkin*  
*Radu Popescu-Zeletin*  
*Luigi Rizzo*  
*Aruna Seneviratne*  
*Otto Spaniol*  
*Chris Blondia*  
*Miklos Boda*  
*Herwig Bruneel*  
*Olga Casals*  
*Tony Ephremides*  
*Andreas Farago*  
*Erol Gelenbe*  
*Mario Gerla*  
*Demetres Kouvasos*  
*Jon Mark*  
*Ioannis Stavrakakis*  
*Jouni Mikkonen*  
*Arne Nilsson*  
*Ramon Puigjaner*  
*Yutaka Takahashi*  
*Phouk Tran-Gia*  
*Jorma Virtamo*  
*Adam Wolisz*  
*Jean-Pierre*  
*Coudreuse*  
*John Luetchford*  
*Guido Petit*  
*Sathya Rao*  
*Joao Rodrigues*  
*Catherine Rosenberg*  
*Jan Slavik*  
*Samir Tohme*  
*Prosper Chemouil*  
*Fabrice Guillemin*

*Gerard Hebuterne*  
*Raif Onvural*  
*Ahmed Tantawy*  
*Karl Lindberger*  
*Marco Marsan*  
*David Hutchison*  
*Ralf Steinmetz*  
*Giovanni Colombo*  
*Serge Fdida*  
*Debasis Mitra*  
*Ulf Koerner*  
*James Roberts*  
*Lyman Chapin*  
*Philippe Owezarski*  
*Bart Steyaert*  
*Stefano Giordano*  
*Samuli Aalto*  
*Norbert Vicari*  
*Patricia Wilcox*  
*Michael Menth*  
*Stefan Koehler*  
*Henning Sanneck*  
*Nedo Celandroni*  
*Francesco Potorti*  
*Vittorio Ghini*  
*Raffaele Perego*  
*Kurt Tutschku*  
*Shoji Kasahara*  
*Oliver Rose*  
*Mathias Duemmler*  
*Kenji Leibnitz*  
*Lars Wolf*  
*Luigi Verri*  
*Marco Di Concetto*  
*Nikolaos Mitrou*  
*Fabrizio Zizza*  
*Roberto Paglino*  
*Giorgio Gallassi*  
*Andrea Bianco*  
*Emilio Leonardi*

Meo Michela  
 Claudio Casetti  
 Renato Lo Cigno  
 Marco Mellia  
 Fabio Neri  
 Gian Paolo Rossi  
 Iakovos Venieris  
 Luigi Musumeci  
 Franco Davoli  
 Andrea Baiocchi  
 Sergio Palazzo  
 Sebastien Ardon  
 Christos Douligeris  
 Leandros Tassioulas  
 Dimitrios Serpanos  
 Michael Paterakis  
 Jeff Capone  
 Elias Manolakos  
 Sophia Tsakiridou  
 Bjorn Landfeldt  
 Jonathan Chan  
 Enzo Mingozzi  
 Dirk Staehle  
 Celia Ramos  
 J. Monteiro  
 Boris Tsybakov  
 Dimitrios Makrakis  
 Michael Devetsikiotis  
 Klara Nahrstedt  
 Denis Collange  
 Philippe Olivier  
 Marinho Barcellos  
 Edmundo Madeira  
 Rene Boel  
 Udo Krieger  
 Illka Norros  
 Villy Baek Iversen  
 Emilio Leonardi  
 Jose R. Gallardo  
 Ioannis Nikolaidis  
 Naoaki Yamanaka  
 Guido H. Petit  
 Masaki Aida  
 Ioannis Lambadaris  
 Kaoru Sezaki  
 Naoto Miyoshi  
 Arata Koike

Hiroyuki Kawano  
 Shigefusa Suzuki  
 Noriaki Kamiyama  
 Naoki Naoki  
 Wakamiya  
 Miki Yamamoto  
 Marcelo Rubinstein  
 Paulo Goncalves  
 Paulo Vidal  
 Artur Ziviani  
 Bo Friis Nielsen  
 Lee Johnston  
 John Cushnie  
 Steven Simpson  
 Laurent Mathy  
 Mark Banfield  
 Ryutaro Kawamura  
 Piet Demeester  
 Bo Friis Nielsen  
 Sebastia Galmes  
 Harry G. Perros  
 Henrik Christiansen  
 Salvador Balle  
 Guillem Femenias  
 Nikolas Mitrou  
 David McDonald  
 Hajer Tounsi  
 Ilhem Lengliz  
 Antonio Loureiro  
 Abdelfattah Belghith  
 Thomas Fuhrmann  
 Farouk Kamoun  
 Jogesh Muppala  
 Manivasakan  
 Rathinam  
 Jean-Jacques Pansiot  
 Thomas Ziegler  
 Guy Pujolle  
 Kave Salamatian  
 Khaldoun Al Agha  
 Ayalvadi Ganesh  
 Sambit Sahu  
 Supratik  
 Bhattacharyya  
 Koenraad Laevens  
 Lars Staalhagen  
 Allan T. Andersen

Dorte Moeller  
 Jana Tuckova  
 Qiang Ren  
 Arunabha Sen  
 Jose M. Barcelo  
 Dominique Delisle  
 Gyorgy Miklos  
 Lloren Cerd  
 Leos Bohac  
 Victor Yau  
 Jianping Pan  
 Yu Zeng  
 Brahim Bensaou  
 Dongmei Zhao  
 Jianping Pan  
 Olivier Bonaventure  
 Jean-Marc Uze  
 Geoffroy Jennes  
 Ravi Mazumdar  
 Ness Shroff  
 Laurent Massoulie  
 Edwin Chong  
 Andre Girard  
 Hossam Abdelbaki  
 William Lau  
 Anton Kuchar  
 Koenraad Laevens  
 Norvald Stol  
 Olav Osterbo  
 Pavel Hofman  
 Laurent Dairaine  
 Saul Pomares  
 Frantisek Krizovsky  
 Antonio Grilo  
 Ragnar Andreassen  
 Khaled Elsayed  
 Aarstad Egil  
 Rui Lopes  
 Teresa Vazao  
 Bjarne E. Helvik  
 Kimon Kontovasilis  
 Peder Emstad  
 Evangelos Zerva



# Table of Contents

## Session 1: Multicasting in Heterogeneous Networks

Multi-criteria Arguments for Improving the Fairness of Layered Multicast Applications Marcelo Dias de Amorim, Otto Carlos M.B. Duarte, and Guy Pujolle .....	1
Network-Driven Layered Multicast with IPv6 Ho-pong Sze and Soung C. Liew .....	11
Analysis of the Requirements for ATM Multicasting Based on Per-PDU ID Assignment Josep Mangues-Bafalluy and Jordi Domingo-Pascual .....	23

## Session 2: Performance of ATM Networks

Sensitivity of ABR Congestion Control Algorithms to Hurst Parameter Estimates Sven A.M. Östring, Harsha Sirisena, and Irene Hudson .....	36
Providing GFR Guarantees for TCP/IP Traffic over APON Access Systems Sašo Stojanovski, Maurice Gagnaire, and Rudy Hoebeke .....	49
Buffer Size Requirements for Delay Sensitive Traffic Considering Discrete Effects and Service-Latency in ATM Switches Steven Wright and Yannis Viniotis .....	61

## Session 3: Packet Switching Node Architectures

Distributed Input and Deflection Routing Based Packet Switch Using Shuffle Pattern Network Thai Thach Bao, Hiroaki Morino, Hitoshi Aida, and Tadao Saito .....	74
A Distributed Dynamic Scheduling Algorithm for a Terabit Multicast Packet Switch Feihong Chen, Necdet Uzun, and Ali N. Akansu .....	85
An Interworking Call Control Solution for a Multidiscipline Switch Pertti Raatikainen and Sami Raatikainen .....	98

## Session 4: Multicasting I

Generic Multicast Transport Services: Router Support for Multicast Applications Brad Cain and Don Towsley.....	108
---	-----

XIV Table of Contents

RP-Based Multicast Receiver Access Control in PIM-SM  
Thomas Hardjono ..... 120

An Algorithm for Multicast with Multiple QoS Constraints  
and Dynamic Membership  
Aiguo Fei and Mario Gerla ..... 132

**Session 5: QoS and Traffic Issues**

Optimal Traffic Partitioning in MPLS Networks  
Esmael Dinan, Daniel O. Awduche, and Bijan Jabbari ..... 144

Impact of the Ethernet Capture Effect on Bandwidth Measurements  
Mats Björkman and Bob Melander ..... 156

**Session 6: Optical Networks**

Statistical Study of the Correlation Between Topology and Wavelength Usage  
in Optical Networks with and without Conversion  
Christian Fenger, Emmanuel Limal, Ulrik Gliese, and Cathal J. Mahon ..... 168

Computing Blocking Probabilities in Multi-class Wavelength Routing Networks  
Sridhar Ramesh, George N. Rouskas, and Harry G. Perros ..... 176

A Comparative Study of Mesh and Multi-ring Designs  
for Survivable WDM Networks  
Thanyaporn Iamvasant, Charoenchai Baworntummarat,  
and Lunchakorn Wuttisittikulkij ..... 189

**Session 7: Multicasting II**

Membership-Insensitive Totally Ordered Multicast: Properties and Performance  
Jerzy Konorski ..... 201

On Finding Feasible Solutions to the Group Multicast Routing Problem  
Ning Wang and Chor Ping Low ..... 213

**Session 8: TCP/IP over ATM**

Analysis of Cell Spacing on a TCP Flow Mapped onto an ATM Connection  
Fabrice Guillemin, Jacqueline Boyer, Olivier Dugeon, and Christophe Mangin.... 228

Mapping of Loss and Delay Between IP and ATM Using Network Calculus  
Tijani Chahed, Gérard Hébuterne, and Caroline Fayet..... 240

**Session 9: Traffic with Long Range Dependencies**

A Self-Similar Point Process Through Fractal Construction Manjunath A. Krishnam, Anand Venkatachalam, and Jeffrey M. Capone.....	252
---	-----

Tail Transitions in Queues with Long Range Dependent Input Tim Daniëls and Chris Blondia.....	264
--	-----

**Session 10: Congestion Control in Broadband Multicast Networks**

An Exact Algorithm for Calculating Blocking Probabilities in Multicast Networks Eeva Nyberg, Jorma Virtamo, and Samuli Aalto.....	275
--	-----

Approximations for Call-Blocking Probabilities in Multirouting Multicasting Multirate Loss Networks Tibor Cinkler and László Ast.....	287
---	-----

Consolidation Algorithm Interoperability in Point-to-Multipoint ABR Services in ATM Networks Naris Rangsinoppamas, Tanun Jaruvitayakovit, Wisitsak Sa-niamsak, Praphan Pavarangkoon, and Prasit Prapinmongkolkarn .....	299
--	-----

**Session 11: QoS and Signalling**

Evaluation of the INSIGNIA Signaling System Seoung-Bum Lee, Gahng-Seop Ahn, Xiaowei Zhang, and Andrew T. Campbell..	311
--	-----

Design and Implementation of RSVP Based on Object-Relationships Martin Karsten .....	325
---	-----

Indirect RSVP for Virtual Cluster Cellular Mobile IP Networks Yu Zeng, Jon W. Mark, and Xuemin Shen.....	338
---	-----

**Session 12: Voice/Video Traffic Modelling**

Aggregation of Markovian Sources: Approximations with Error Control Marco Conti, Silvia Ghezzi, and Enrico Gregori.....	350
--	-----

Multi-scaling Models of Sub-frame VBR Video Traffic Iraj Saniee, Arnold Neidhardt, Onuttom Narayan, and Ashok Erramilli .....	362
--	-----

An Accurate Closed-Form Formula to Calculate the Dejittering Delay in Packetised Voice Transport D. De Vleeschauwer, G.H. Petit, B. Steyaert, S. Wittevrongel, and H. Bruneel ....	374
--	-----

**Session 13: IP and ATM Interworking**

Interworking of B-ISDN Signaling and Internet Protocol Muneyoshi Suzuki .....	386
Mapping an Internet Assured Service on the GFR ATM Service Fernando Cerdán and Olga Casals .....	398
Real-Time Cell Arrival Sequence Estimation and Simulator for IP/ATM Networks Hiroshi Saito, Toshiaki Tsuchiya, Gyula Marosi, Gyorgy Horvath, Peter Tatai, and Shoichiro Asano .....	410

**Session 14: Single-Server Queues**

Computing Stochastic Bounds for the Tail Distribution of an M/GI/1 Queue Pierre L. Doullet, André-Luc Beylot, and Monique Becker .....	423
Analysis of Packet Delay in a GI-G-1 Queue with Non-preemptive Priority Scheduling Joris Walraevens, Bart Steyaert, and Herwig Bruneel .....	433
New Results on the Numerical Stability of the Stochastic Fluid Flow Model Analysis Markus Fiedler and Holger Voos .....	446

**Session 15: Real-Time Traffic Management**

QoS Support for Real-Time Applications Using the Integration of RSVP/Intserv and Diffserv: A Testbed Experiment Seong-Ho Jeong, Myung Choi, Randal Abler, Henry Owen, John Copeland, and Joachim Sokol .....	458
Efficient End-to-End Transport of Soft Real-Time Applications Zoe Antoniou and Ioannis Stavrakakis .....	470
Real-Time Traffic Transmission over the Internet Marco Furini and Don Towsley .....	483

**Session 16: High Speed Wireless Access**

High Speed Wireless Internet Access: Combination of MPLS and BRAN HIPERLAN/2 Technologies Kallikratidas Nitsos, Tasos Dagiuklas, and Hamed Al-Raweshidy .....	495
Effect of Turn-Around Times on the Performance of High Speed Ad-hoc MAC Protocols Ajay Chandra V. Gummalla and John O. Limb .....	507

Experimental Investigation of the Effects of Delay and Error Rate on Throughput and Performance of ATM Satellite Links Serving LAN Islands Sufian Yusef and Caroline Stange .....	518
---	-----

### **Session 17: Differentiated Services**

Dynamic Algorithms with Worst-Case Performance for Packet Classification Pankaj Gupta and Nick McKeown.....	528
--	-----

Intelligent Traffic Conditioners for Assured Forwarding Based Differentiated Services Networks Biswajit Nandy, Nabil Seddigh, Peter Pieda, and Jeremy Ethridge .....	540
--	-----

DiffServ in the Web: Different Approaches for Enabling Better Services in the World Wide Web Hartmut Ritter, Thorsten Pastoors, and Klaus Wehrle.....	555
---	-----

### **Session 18: Internet Performance and Control**

Performance of Short TCP Transfers Chadi Barakat and Eitan Altman .....	567
--	-----

Performance Study of Satellite-Linked Web Caches and Filtering Policies Xiao-Yu Hu, Pablo Rodriguez, and Ernst W. Biersack .....	580
---	-----

Distributed Control System for Low Priority Controllable Traffic in Packet Switched Backbone Networks Kimmo Pulakka and Jarmo Harju.....	596
--	-----

### **Session 19: Residential Access Networks**

Application-Independent End-to-End Security in Shared-Link Access Networks José C. Brustoloni and Juan A. Garay .....	608
--	-----

Fair Efficient Call Admission Control Policies for Heterogeneous Traffic Streams in a Packet Routing Server Using the DTM Technology Chih-Jen Chang and Arne A. Nilsson.....	620
--	-----

### **Session 20: QoS Negotiation**

An Agent-Based Framework for Large Scale Internet Applications Mamadou Tadiou Kone and Tatsuo Nakajima .....	632
---	-----

Benchmarking of Signaling Based Resource Reservation in the Internet István Cselényi, Gábor Fehér, and Krisztián Németh.....	643
---	-----

XVIII Table of Contents

**Session 21: Resource Management**

Kalman and Neural Network Approaches  
for the Control of a VP Bandwidth in an ATM Network  
Raphaël Feraud, Fabrice Clérot, Jean-Louis Simon, Daniel Pallou,  
Cyril Labbé, and Serge Martin..... 655

Fairness and Aggregation: A Primal Decomposition Study  
André Girard, Catherine Rosenberg, and Mohammed Khemiri ..... 667

**Session 22: QoS in Wireless Networks**

On Learning and the Quality of Service in a Wireless Network  
Tiina Heikkinen ..... 679

Distributed Fair Bandwidth Allocation of a Wireless Base Station  
György Miklós and Sándor Molnár ..... 689

Fast Approximate Algorithms for Maximum Lifetime Routing  
in Wireless Ad-hoc Networks  
Jae-Hwan Chang and Leandros Tassiulas..... 702

**Session 23: Delay Bounds**

Performance Evaluation of Resource Division Policies  
for PGPS Scheduling in ATM Networks  
Amr S. Ayad, Khaled M.F. Elsayed, and Mahmoud T. El-Hadidi ..... 714

Worst-Case Deterministic Delay Bounds  
for Arbitrary Weighted Generalized Processor Sharing Schedulers  
Robert Szabó, Peter Barta, Felician Németh, and Jozsef Bíró..... 727

Deterministic End-to-End Delay Bounds in an Accumulation Network  
Guillaume Urvoy, Gérard Hébuterne, and Yves Dallery ..... 740

**Session 24: Internet Protocols**

Performance Analysis of IP Switching and Tag Switching  
Gargi Banerjee, Rob Rosenberry, and Deepinder Sidhu..... 752

A Distributed Mechanism for Identification and Discrimination  
of Non-TCP-friendly Flows in the Internet  
Thomas Ziegler and Serge Fdida ..... 763

**Session 25: Wireless LAN**

New Handoff Strategies in Microcell/Macrocell Overlaying Systems Selma Boumerdassi and André-Luc Beylot.....	776
---	-----

Dynamic IEEE 802.11: Design, Modeling and Performance Evaluation Federico Cali, Marco Conti, and Enrico Gregori.....	786
---	-----

TCP/IP over the Bluetooth Wireless Ad-hoc Network Niklas Johansson, Maria Kihl, and Ulf Körner.....	799
--	-----

**Session 26: Traffic Control for Quality of Service**

A Scheme for Time-Dependent Resource Reservation in QoS-Enabled IP Networks Roberto Canonico, Simon Pietro Romano, Mauro Sellitto, and Giorgio Ventre ....	811
--	-----

Impact of "Trunk Reservation" on Elastic Flow Routing Sara Oueslati-Boulahia and James W. Roberts.....	823
---	-----

**Session 27: Routing**

New Distributed Multicast Routing and Its Performance Evaluation Takuya Asaka, Takumi Miyoshi, and Yoshiaki Tanaka .....	835
---	-----

Distance-Vector QoS-Based Routing with Three Metrics Luís Henrique M.K. Costa , Serge Fdida, and Otto Carlos M.B. Duarte.....	847
--	-----

On Shortest Path Problems with "Non-Markovian" Link Contribution to Path Lengths Arunabha Sen, K. Selçuk Candan, Afonso Ferreira, Bruno Beauquier, and Stephane Perennes.....	859
--	-----

**Session 28: Quality of Service for Video and Multimedia**

Adaptive QoS Platform in Multimedia Networks Mahmoud Sherif, Ibrahim Habib, Mahmoud Naghshineh, and Parviz Kermani ....	871
--	-----

Quality-of-Service (QoS) in Heterogeneous Networks: CLIP, LANE and MPOA Performance Test Kai-Oliver Detken.....	883
---	-----

Experiments with Dynamic Multiplexing and UPC Renegotiation for Video over ATM Maria Teresa Andrade and Artur Pimenta Alves .....	895
---	-----

**Session 29: Call Admission Control in Cellular Systems**

The Study of Burst Admission Control in an Integrated Cellular System  
Jie Zhou, Yoshikuni Onozato, and Ushio Yamamoto..... 908

Model and Optimal Call Admission Policy in Cellular Mobile Networks  
Amine Berqia and Noufissa Mikou ..... 920

Measurement-Based Pre-assignment Scheme with Connection-Level QoS Support  
for Multiservice Mobile Networks  
Xiaoyuan Luo, Ian Thng, Bo Li, and Shengming Jiang..... 932

**Session 30: Resource Allocation**

QoS Rewards and Risks: A Multi-market Approach to Resource Allocation  
Errin W. Fulp and Douglas S. Reeves ..... 945

Spare Capacity Planning for Survivable Mesh Networks  
Adel Al-Rumaih, David Tipper, Yu Liu, and Bryan A. Norman..... 957

A Cooperative Game Theory Approach to Resource Allocation in Wireless ATM  
Networks  
Xinjie Chang and Krishnappa R. Subramanian ..... 969

**Author Index**..... 979



# Multi-criteria Arguments for Improving the Fairness of Layered Multicast Applications<sup>\*</sup>

Marcelo Dias de Amorim<sup>1,2</sup>, Otto Carlos M. B. Duarte<sup>2</sup>, and Guy Pujolle<sup>1</sup>

<sup>1</sup> PRiSM Laboratory, University of Versailles  
45, Avenue des Etats-Unis – 78035 – Versailles – France  
`{amorim,gp}@prism.uvsq.fr`

<sup>2</sup> GTA/COPPE/EE – Federal University of Rio de Janeiro  
P.O. Box 68504 – 21945-970 – Rio de Janeiro – RJ – Brazil  
`otto@gta.ufrj.br`

**Abstract.** In this paper, we address the problem of choosing the most adequate rates for the different layers in multicast distribution of multi-layered applications. We argue that depending on the type of the application that is being carried out, we face a number of problems to decide what are the most adequate bounds on the parameters to achieve the best global quality. We propose then multi-decision criteria that improve the global aggregated quality of the communication in the multicast session. We consider the quality of the application at the receiver, the bandwidth utilization, and the satisfaction index to measure the combined user/network quality of the communication.

## 1 Introduction

For the transport of multimedia applications through multicast sessions with heterogeneous receivers, the use of multi-layered coding has shown to be a worthwhile approach. In such a technique, the source is coded in a base layer and in one or more enhancement layers and the quality of the presentation depends on the number of layers that are played. Multi-layered coding has been applied to improve the fairness of a wide variety of multicast applications, mainly for video distribution [1,2,3,4,5,6,7,8,9,10]. This approach is interesting in multicast communications because destinations with different receiving capacities (available bandwidth, local resources, etc) can choose the layers they are able to receive.

Multi-layered applications over multicast networks require that new mechanisms are embedded in the communication protocol. One of the most difficult problems is to choose the metrics that should be considered by the algorithm that computes the rates of the video layers in order to maximize the global fairness of the application. This is a strategic task because of the subjective decision of which metric should be prioritized. For instance, in [5,8], video rates are calculated based only on the degree of video degradation at the receivers, i.e.,

---

<sup>\*</sup> This work has been supported by CAPES, COFECUB, CNPq, UFRJ, CNRS, and UVSQ.

the difference between the required and the received video rates. In this paper, we propose a set of other potential metrics and argue that one of them should be prioritized according to the type of the application. In our scenario, the application runs over a rate-based adaptive multicast session, where the receivers periodically send control packets to the source containing information about the video rate they want (or are able) to receive [5]. Based on these packets, the source computes the most appropriate rates to optimize the quality of the communication. However, it must be established what means a good global quality. We propose a first approach to this problem by considering the bandwidth utilization, the satisfaction index, and the degree of degradation at the receivers as the metrics to be used by the algorithm to compute the layers. We show that all metrics cannot be optimized at the same time and give some examples showing the dependency between the type of the application and the metric to be prioritized.

The following of this paper is organized in four sections. In Section 2, we present some multi-layered multicast schemes proposed in the literature. In Section 3, we discuss the limitations of using a single metric to compute the rates of the layers. The use of multiple metrics and how these metrics react to fluctuating network congestion are proposed in Section 4. Finally, Section 5 concludes this paper.

## 2 Multi-layered Multicast Schemes

Different strategies have been used to provide quality of service to applications in networks that offer no guarantees such as the Internet. One approach is to apply resource reservation, in which resources are set aside for a particular flow [11]. A more recent technique is to introduce adaptability in the applications [5,6,7,12,13,14,15]. In such a case, applications are kept informed about available network resources and dynamically adapt their transmission rates to the current state of the network. This is particularly interesting in heterogeneous multicast communication in which data must be delivered to a number of destinations with different receiving capacities.

Most adaptive multimedia schemes have used multi-layered coding as the major supporting mechanism to improve the fairness of the applications. The receiver-driven layered multicast (RLM) [7] is a rate-adaptive protocol for the distribution of layered video using multiple IP-multicast groups. The source generates a certain number of video layers and transmits each layer in a different multicast session. The receivers subscribe to the number of groups they will. The communication is dynamic in the way receivers can dynamically join or leave groups. However, they are limited to the layers the source decides to transmit. Since the receivers do not have any influence on the way the rates of each layer are calculated, these values may not be suited to yield good video quality at the destinations and reasonable network utilization.

The Layered Video Multicast with Retransmission (LVRM) [9] has been proposed for the distribution of layered video over the Internet. LVRM deploys an

error recovery scheme using retransmissions with upper bound on recovery time and adaptive playback point. LVRM also implements adaptation to network congestion and heterogeneity using a hierarchical rate control mechanism. In this mechanism, the information about network congestion is exchanged among the sender, the receivers, and some agents in the network. This scheme has the advantage of scalability since each network component stores only the information relevant to itself.

In [5], Vickers et al. propose a rate-based control mechanism in which a number of feedback control packets containing the current network congestion state are exchanged between the source and the receivers. Based on the content of these packets, the source estimates the number of video layers to transmit and their respective rates. In this scheme, the source periodically multicasts control packets to the receivers. As these packets traverse the network, intermediate nodes mark them with the amount of available bandwidth. Each receiver copies the packet's content into a returning feedback control packet and sends it back to the source. To avoid feedback implosion at the source, intermediate nodes implement an algorithm to merge the feedback packets returned by the receivers.

The single-loop packet-merging algorithm proposed in [8] is an algorithm to improve fairness for multicast distribution of multi-layered video. The control information scheme is similar to the approach proposed in [5]. However, intermediate nodes perform packet merging in only one loop and use the concept of virtual layers. With virtual layers, the nodes consider that the source is able to generate more than the actual number of layers. In several circumstances, this leads to an important improvement in the global video quality at the receivers since a reduced number of merging procedures are performed and consequently relevant information are not lost.

All of the above techniques have shown to improve the fairness of multicast applications. However, these algorithms consider either the user or the network when computing the video layers, and we argue that both of them should be taken into account.

### 3 Application-Dependent Quality

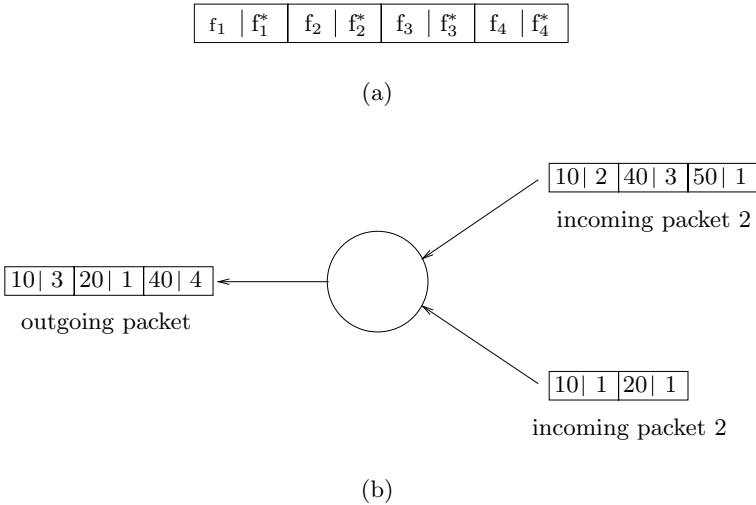
In this section, we evaluate the need for other metrics when computing the rates of the video layers other than the global video degradation ( $\Delta$ ), which is the aggregate of all local degradations, given by the difference between the required video rate ( $r_i$ ) and the actual video rate ( $r_i^r$ ) received by destination  $i$ ,

$$\Delta = \sum_{i=1}^N \delta_i, \quad (1)$$

where  $\delta_i = (r_i - r_i^r)$ ,  $r_i \geq r_i^r$ , and  $N$  is the number of receivers in the session.

We consider in this paper a feedback control scheme similar to the one presented in [5], where feedback control packets are transmitted by the destinations to the source to indicate the video rate they want to receive. These packets are

combined at intermediate nodes to avoid feedback implosion at the source. After a merging procedure, each resulting feedback control packet contains a number of entries that store a video rate ( $f_i$ ) and the number of destinations that want to receive this rate ( $f_i^*$ ), as shown in the example of Fig. 1(a). The number of entries in the feedback control packet corresponds to the number of video layers the source can transmit. When performing a merging procedure, the node looks for the entries that must be discarded according to the merging algorithm and creates a new feedback packet to be sent to the source. Fig. 1(b) shows an example of the merging procedure using the single-loop algorithm for two incoming feedback control packets. In this example, the algorithm has to discard two entries since the two incoming packets together have five entries and the source is supposed to transmit no more than three layers. The first entry of the incoming packet 1 is then combined to the first entry of the incoming packet 2 and entries 2 and 3 of the incoming packet 1 are merged.<sup>1</sup>



**Fig. 1.** (a) Feedback control packet and (b) an example of the merging procedure for a three-layer video.

The algorithms proposed in [5,8], however, are based only on the video degradation at the receivers. In this paper, we propose the use of other metrics when performing the merging procedure. Consider for example a multicast session with one sender and four receivers as shown in Fig. 2. In this scenario, suppose that the source transmits two video layers and that the merging algorithm uses only the global video degradation at the receivers to perform the merging procedure.

<sup>1</sup> For further details on the merging algorithms, please refer to [5,8].

Since the rate of the base layer is limited by the slowest receiver, in our example the source transmits the two video layers at  $L_1=1\text{Mbps}$  and  $L_2=1\text{Mbps}$ . Note in Table 1 that these values lead to the lowest global video degradation.

**Table 1.** Video degradation for layer 2 transmitting at 1 and 2.9 Mbps.

Receiver	$L_2=1\text{Mbps}$			$L_2=2.9\text{Mbps}$		
	required	received	degradation	required	received	degradation
$r_1$	1	1	0	1	1	0
$r_2$	2	2	0	2	1	1
$r_3$	2	2	0	2	1	1
$r_4$	3.9	2	1.9	3.9	3.9	0

Global degradation

1.9 Mbps

2 Mbps

Consider now that the capacity of receiver 4 changes from 3.9 to 4.1Mbps. The new video layers that result in the lowest global degradation are  $L_1=1\text{Mbps}$  and  $L_2=3.1\text{Mbps}$ , and the correspondent global video degradation is  $\Delta=2\text{Mbps}$ . The possible value for the video layers are shown in Table 2. Note that for a variation of 5% in the receiving capacity of receiver 4, the video layer 2 has increased its rate of 310%, which results in a substantial growth of the bandwidth utilization. Moreover, in the first scenario there were three destinations receiving 100% of the required bandwidth (and only one destination receiving  $\sim 50\%$ ), while in the second scenario only two receivers are supplied with 100% and two receivers with 50%. It is quite intuitive that it is better to maintain  $L_1$  and  $L_2$  at 1Mbps in order to have lower bandwidth utilization even if the global video degradation is greater than the optimum value.

**Table 2.** Video degradation for layer 2 transmitting at 1 and 3.1 Mbps.

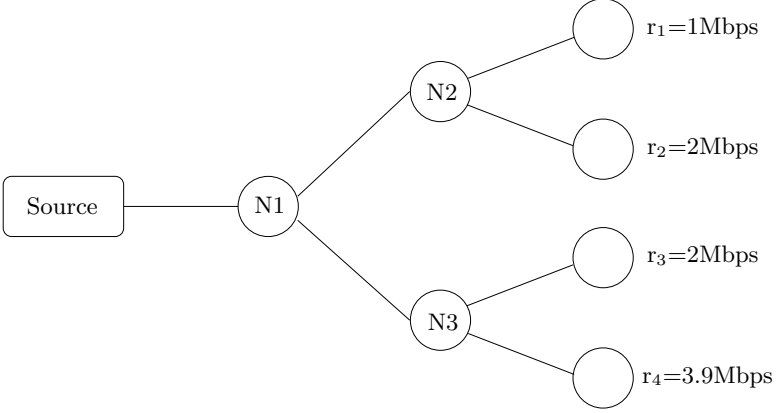
Receiver	$L_2=1\text{Mbps}$			$L_2=3.1\text{Mbps}$		
	required	received	degradation	required	received	degradation
$r_1$	1	1	0	1	1	0
$r_2$	2	2	0	2	1	1
$r_3$	2	2	0	2	1	1
$r_4$	4.1	2	2.1	4.1	4.1	0

Global degradation

2.1 Mbps

2 Mbps

In next section we will present a qualitative analysis on the use of different metrics to improve the global quality of the application. We propose the global degradation, the bandwidth utilization, and the user satisfaction as the metrics to be used by the algorithm in order to improve the quality of the communication.



**Fig. 2.** Example of multicast session.

## 4 Multi-criteria Multicast

It is likely that distinct applications have different requirements about the metrics to be prioritized by the algorithm that computes the video layers. For example in a pay-per-view video service, it is important that the subscribers receive the maximum possible video quality even if the server rejects new calls. In video surveillance, maximizing the video quality may not be the main objective, but the reduction of the allocated network resources. On another side, broadcast advertisements may require that the data are received by a number of destinations as large as possible. The use of different criteria may then be a solution to the problem of multicast delivery of multi-layered applications if it is applied a scheme of weighted parameters, in which each one of the proposed metrics is prioritized depending on the application that is being carried out.

To introduce other metrics, some extensions must be made in the feedback control mechanism. A variation of the feedback control packet must contain other information such as the video degradation and the satisfaction index at the receivers. Besides the video rate and the number of destinations that want to receive this rate (as in the original packet), each entry of the new control packet has two other fields: the video degradation and the satisfaction index.

When performing the merging procedure, the node looks for the entries that have to be discarded and adds its correspondent degradation to the degradation of the entry it is going to be combined with. Consider the same topology of the example shown in Fig. 2 in which the source transmits two video layers. We denote  $L_i^j$  the value of  $f_i$  at the network element  $j$  and  $\delta_i^k$  the video degradation correspondent to the rate  $f_i$  at the network element  $k$ . When generating a feedback control packet, receiver  $d_i$  sets the field  $\delta_1^{d_j}$  to 0 (no degradation since no merging procedure has been performed) and sends the packet to its upstream node. The feedback control mechanism performs the following steps:

At node  $N_2$  (no merging procedure):

$$\begin{array}{ll} L_1^{d_1} \rightarrow L_1^{n_2} & \delta_1^{d_1} \rightarrow \delta_1^{n_2} \\ L_1^{d_2} \rightarrow L_2^{n_2} & \delta_1^{d_2} \rightarrow \delta_2^{n_2} \end{array}$$

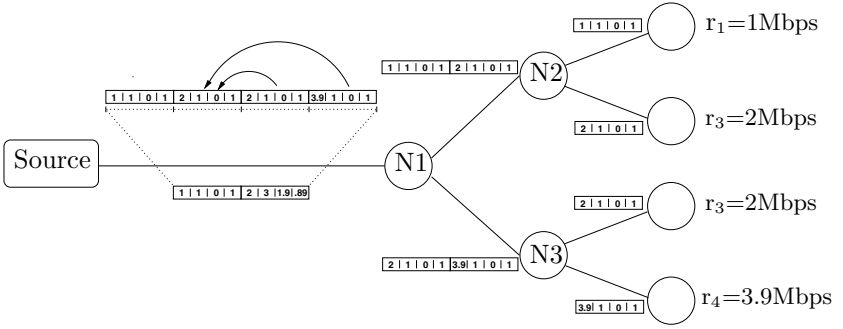
At node  $N_3$  (no merging procedure):

$$\begin{array}{ll} L_1^{d_3} \rightarrow L_1^{n_3} & \delta_1^{d_3} \rightarrow \delta_1^{n_3} \\ L_1^{d_4} \rightarrow L_2^{n_3} & \delta_1^{d_4} \rightarrow \delta_2^{n_3} \end{array}$$

At node  $N_1$ :

$$\begin{array}{ll} L_1^{n_2} \rightarrow L_1^{n_1} & (\delta_1^{n_2} + \delta_2^{n_2} + (L_2^{n_2} - L_1^{n_2})) \rightarrow \delta_1^{n_1} \\ L_1^{n_3} \rightarrow L_2^{n_1} & (\delta_1^{n_3} + \delta_2^{n_3} + (L_2^{n_3} - L_1^{n_3})) \rightarrow \delta_2^{n_1} \end{array}$$

In this way, the information about the video degradation is carried through the multicast tree and the source can estimate the global video degradation by computing  $\Delta = \sum_{i=1}^E \delta_i$ , where  $E$  is the number of entries in the packet. Fig. 3 shows the numerical results after each one of the steps.



**Fig. 3.** The merging steps at each one of the nodes.

The degree of bandwidth utilization is computed directly at the source by combining the video rates and their respective number of requests. For a source transmitting  $C$  layers, we call weighted bandwidth allocation ( $\Omega$ ) the estimation of the bandwidth allocated throughout the multicast tree,

$$\Omega = \frac{\sum_{i=1}^C (f_i^* \times f_i)}{\sum_{i=1}^C f_i^*}. \quad (2)$$

The satisfaction index ( $\nu$ ) measures the average degree of satisfaction in a group of users. It corresponds to the percentage of the received video with regard to the requested video rate. Consider a scenario in which  $(M-1)$  entries  $\{[f_2, f_2^*, \delta_2, \nu_2] \dots [f_M, f_M^*, \delta_M, \nu_M]\}$  are combined to entry  $[f_1, f_1^*, \delta_1, \nu_1]$ . The resulting entry is given by  $[f_1', f_1'^*, \delta_1', \nu_1']$ , where

$$f'_1 = f_1, \quad (3)$$

$$f_1^{*'} = \sum_{i=1}^M f_i^*, \quad (4)$$

$$\delta_1' = \sum_{i=1}^M \delta_i + \sum_{i=2}^M (f_i - f_1) f_i^*, \quad (5)$$

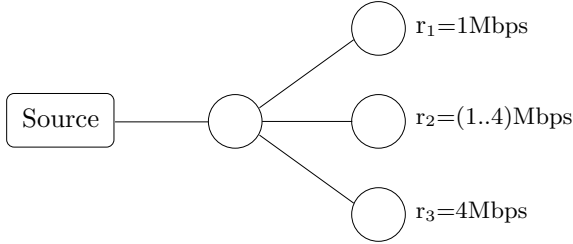
and

$$\nu_1' = \frac{\sum_{i=1}^M (f_i^* \times \nu_i')}{\sum_{i=1}^M f_i^*}, \quad (6)$$

where, for  $j = 2 \dots M$ ,

$$\nu_j' = \frac{f_1}{f_j} \times \nu_j. \quad (7)$$

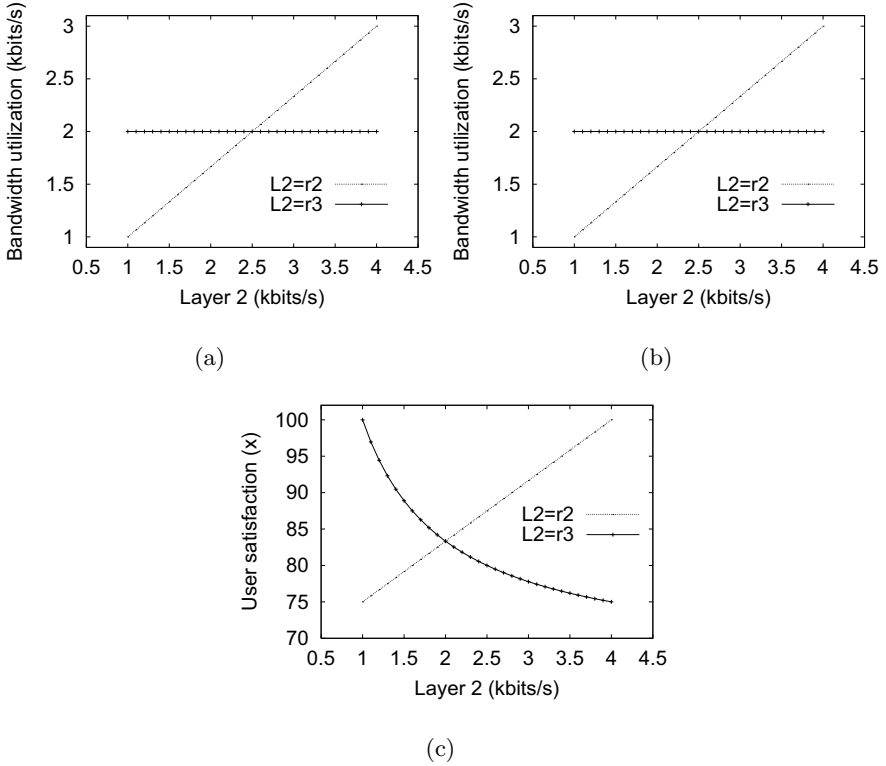
To illustrate that the three metrics proposed in this paper have different comportment and that the combination of them is important to improve the global quality of the communication, we show a simple example of a multicast session with one sender transmitting two video layers and three receivers as shown in Fig. 4. Receivers  $d_1$  and  $d_3$  have fix receiving capacities ( $r_1=1\text{Mbps}$  and  $r_3=4\text{Mbps}$ , respectively), and the available bandwidth in the path to destination  $d_2$  varies from  $r_1$  to  $r_3$ .



**Fig. 4.** Example topology.

Fig. 5 shows how the metrics respond to the different possible values of the video layers, i.e., when  $\{L_1 = r_1; L_2 = r_2\}$  and  $\{L_1 = r_1; L_2 = r_3\}$ . The global degradation for the two possibilities are show in Fig. 5(a). Similarly, Fig. 5(b) and Fig. 5(c) depict, respectively, the weighted bandwidth utilization and the estimated satisfaction index. Note that the ideal is to have the minimum global degradation, minimum bandwidth utilization, and the maximum satisfaction





**Fig. 5.** (a) Global video degradation, (b) bandwidth utilization, and (c) satisfaction index.

index. However, the example shows that these values cannot be simultaneously obtained. In fact, the type of the application must be taken into account when the algorithm to compute the layers is being performed.

## 5 Conclusions

We have proposed in this paper the use of multiple metrics to compute the most adequate rates for multicast multi-layered applications. We consider the global degradation at the receivers, the bandwidth utilization, and the satisfaction index. We have also argued that one of these metrics should be prioritized to improve the quality of the communication. We showed how the metrics are calculated and how they respond to variations on the network congestion. The proposed criteria lead to a significant improvement on the quality of the application when compared with the classical approaches that consider only the degradation at the destinations.

## References

1. D. Saporilla and K. W. Ross, "Optimal streaming of layered video," in *IEEE Infocom*, (Tel-Aviv, Israel), Mar. 2000.
2. T. Wong, R. Katz, and S. McCanne, "A preference clustering protocol for large-scale multicast applications," in *Networked Group Communication - First International COST264 Workshop, NGC'99*, (Pisa, Italy), Nov. 1999.
3. S. Sarkar and L. Tassiulas, "Distributed algorithms for computation of fair rates in multirate multicast trees," in *IEEE Infocom*, (Tel-Aviv, Israel), Mar. 2000.
4. D. Rubenstein, J. Kurose, and D. Towsley, "The impact of multicast layering on network fairness," in *ACM Sigcomm*, (Cambridge, Massachusetts, USA), Sept. 1999.
5. B. J. Vickers, C. Albuquerque, and T. Suda, "Adaptive multicast of multi-layered video: Rate-based and credit-based approaches," in *IEEE Infocom*, (San Francisco, CA, USA), Mar. 1998.
6. B. J. Vickers, C. Albuquerque, and T. Suda, "Source adaptive multi-layered algorithms for real-time video distribution," tech. rep., Information and Computer Science – University of California at Irvine, 1999.
7. S. McCanne, V. Jacobson, and M. Vitterli, "Receiver-driven layered multicast," in *ACM SIGCOMM*, (Stanford, CA, USA), Aug. 1996.
8. M. D. de Amorim, O. C. M. B. Duarte, and G. Pujolle, "Single-loop packet merging for receiver oriented multicast multi-layered video," in *International Conference in Computer Communication*, (Tokyo, Japan), Sept. 1999.
9. X. Li, S. Paul, and M. H. Ammar, "Layered video multicast with retransmissions (LVRM): Evaluation of hierarchical rate control," in *IEEE Infocom*, (San Francisco, CA, USA), Mar. 1998.
10. X. Li, M. H. Ammar, and S. Paul, "Video multicast over the internet," *IEEE Network Magazine*, Mar. 1999.
11. L. Zhang, S. Deering, D. Estrin, S. Shenker, and D. Zappala, "RSVP: A new resource reservation protocol," *IEEE Network Magazine*, Sept. 1993.
12. B. J. Vickers, M. Lee, and T. Suda, "Feedback control mechanisms for real-time multipoint video services," *IEEE Journal on Selected Areas in Communications*, vol. 15, no. 3, Apr. 1997.
13. J. C. Bolot and T. Turetti, "Experience with control mechanisms for packet video in the internet," *ACM Computer Communication Review*, vol. 28, no. 1, Jan. 1998.
14. D. G. Wadington and D. Hutchison, "A general model for QoS adaptation," in *IEEE/IFIP International Workshop on Quality of Service (IWQoS)*, (Napa, CA, USA), May 1998.
15. C. Diot, C. Huitema, and T. Turetti, "Multimedia applications should be adaptive," in *HPCS'95*, (Mystic, CN), Aug. 1995.

# Network-Driven Layered Multicast with IPv6

Ho-pong Sze and Soung C. Liew

Department of Information Engineering, The Chinese University of Hong Kong, Shatin,  
N.T., Hong Kong  
{hpsze8, soung}@ie.cuhk.edu.hk

**Abstract.** Receiver-driven Layered Multicast (RLM) has previously been proposed to distribute video over the best-effort heterogeneous IP networks. Although RLM can avoid network overloading in multicast, its rate control mechanism performs poorly under bursty traffic conditions, which is the characteristic of today's Internet. In this paper, we propose a new scheme called Network-Driven Layered Multicast (NLM) which makes use of IPv6, the next generation IP. Although IPv6 is not currently used in the Internet and Mbone, it will be undoubtedly adopted in the near future due to the running out of IPv4 address space. With the new priority-dropping feature offered by IPv6, the rate adaptation control process in our scheme is much simplified and the reception rate at receivers can be optimized even under bursty background traffic conditions.

## 1 Introduction

As the capacity of the Internet increases, various multimedia network applications that provide audio-visual services such as VOD and video conferencing are becoming increasingly popular. Due to the heterogeneous nature of the Internet, which interconnects networks of diverse regions and different technologies together, high-quality video distribution by multicast is particularly challenging. In a video multicast system, some users may join in from high-speed FDDI networks while others may participate by slow dial-up telephone lines. The main issue here is that the sender can only transmit packets at a single rate. If the video is transmitted in a high bit rate, dial-up users will not be able to participate. But if it is sent out at modem speed, users from high-speed network will then suffer from low-quality service.

To deal with this rate control problem, some researchers have proposed shifting of rate adaptation task from the sender to the network or receivers. One of such approaches is the deployment of video gateways [1, 9] at the network which transcode a high-quality video multicast stream into a low bit rate one before the stream flows into a low-speed network so that users can still participate with their limited bandwidth. However, this approach has two shortcomings. First of all, in order to perform the stream transcoding, the gateway must be able to understand the semantic content of the stream. This requires additional efforts on the network. Moreover, the performance of this approach is highly dependent on the placement of gateways at appropriate locations. But these locations may not be available because of security or resource restriction.

An alternative approach is known as Layered Video Multicast [9]. This approach attempts to accommodate network heterogeneity by tuning the reception rate at the receiver end to match the network capacity. The reception rate can be controlled by subscribing to a different number of video layers. As this approach is receiver-driven, each individual user is able to optimize to a different transmission rate.

Apart from the network heterogeneity problem, the Internet is based on the best-effort network-service model. Since there is no guarantee that packets can reach the destination or be delivered on time and in order, the Internet itself is not suitable for real-time traffic, which is subject to strict loss rate and latency constraints. To cite an example, MPEG is a widely used video coding scheme, but it is not designed for transmission in an unreliable channel like the Internet. An MPEG video sequence typically consists of three types of frames: I-frames, P-frames and B-frames. In order to achieve high compression efficiency, motion compensation technique is used to remove temporal redundancy in successive frames when coding the P-frames and B-frames. While I-frames are independently decodable, other types of frames can only be decoded from its reference frames. Therefore, a packet loss in a reference frame during transmission not only introduces an error in the current frame but will also propagate additional errors to the subsequent frames which will refer to it for decoding. As a result, the perceived video quality at the receiver is severely degraded. Several techniques such as forward error correction (FEC) and introduction of more intra-frames in a video sequence are proposed to control the packet loss and error propagation problem, but none of them are efficient as extra bandwidth is required to transmit the redundant packets. Moreover, the reliability of these techniques depends on the degree of redundancy added, which should be determined by estimating channel characteristics. However, the time-variant and dynamic traffic condition in the Internet makes it hard to attain an accurate estimate.

Alternatively, a more feasible solution to overcome packet loss in multimedia streaming is to retransmit the packet when loss is detected. Although retransmission-based error recovery has been criticized as being ineffective in wide-area real-time communications because of the incursion of extra latency in retransmitted packets, effective recovery is still possible if 1) packet loss can be detected as soon as possible; and 2) an adequate amount of buffering is introduced at receiver to delay the play-out time of presentation. Recent research also suggested that retransmission-based error control work well in real-time multicast if lost packets can be re-sent immediately upon request [3].

In this paper, we propose a new video multicast approach which can overcome network heterogeneity and packet loss simultaneously. To do so, we adopt IPv6 in the layered video multicast scheme. The new *Traffic Class* field in IPv6 provides an excellent priority-dropping feature to distribute layered video in the IP networks without the effort from sender or individual users for adaptation. Although recent research results suggested that priority dropping of packets without any additional error control measures may not have a significant performance improvement over uniform dropping in layered video distribution under smooth background traffic, it does give a modest gain, especially under bursty traffic conditions like the Internet [10]. By using our loss recovery scheme, error control can be even more effective and a better performance can be achieved. The paper proceeds as follows: Section 2 gives an overview of RLM and IPv6. Section 3 presents our proposed scheme known as Network-Driven Layered Multicast with IPv6 (NLM). Section 4 analyses the performance of the new scheme. Section 5 concludes our work.

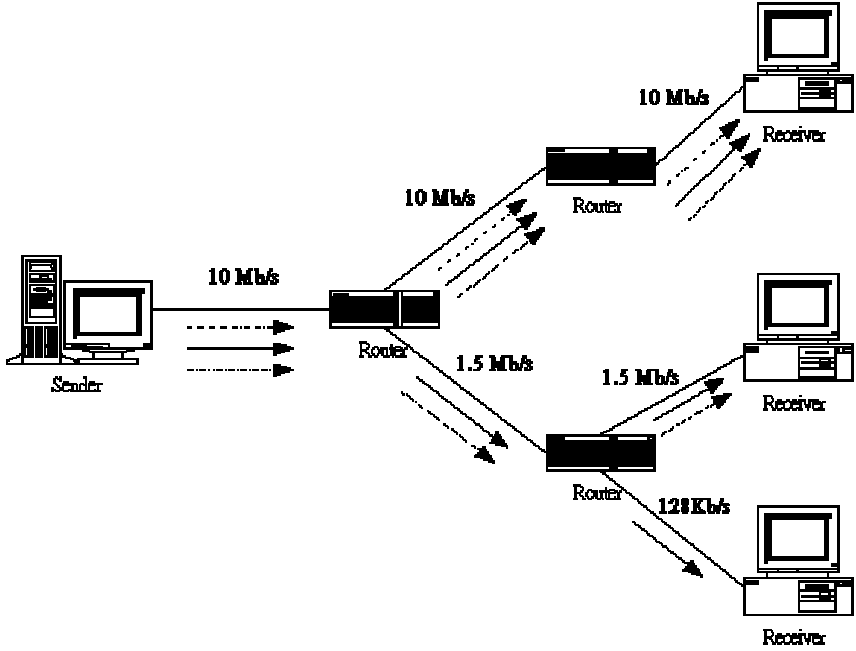


Fig. 1. An Example of RLM

## 2 Background

### 2.1 Receiver-Driven Layered Multicast (RLM)

As mentioned in the last section, one approach for real-time video multicast on the IP networks is the use of layered video multicast. One typical example of such schemes is Receiver-driven Layered Multicast [2]. In this scheme, the video source is first encoded into multiple layers. The base layer provides the basic quality of video and each higher enhancement layer progressively improves the signal quality. Depending on the compression schemes, a video sequence can be layered by picture resolution, quantization factor and frame rate. At the sender, each layer is transmitted to a distinct multicast group. In turn, based on its network capacity, each receiver subscribes to the number of layers which can obtain the best possible quality. As the bandwidth between the source and receiver may vary from time to time, dynamic rate adaptation is carried out by the receiver joining/leaving multicast groups during the subscription period. When congestion is detected in the network, a receiver drops the highest enhancement layer to reduce reception rate and hence alleviate congestion. When congestion frees up, it subscribes to an additional layer. Figure 1 shows an example of RLM. Three layers of a video are represented by different types of arrows. By subscribing to different number of layers according to available bandwidth, all receivers in the group are able to optimize their reception rates.

While network congestion can be easily detected by high packet-loss rate at the receiver, the detection of unused network capacity is more difficult. In RLM, a receiver tries to detect extra available bandwidth by periodical probing. That is, in every predefined period, the receiver subscribes to an additional layer to observe the reception quality. If the quality is degraded rather than improved, this indicates network congestion results and hence the receiver should immediately drop the newly added layer. How often should the receiver do the probing is an issue. A recent research work used the decrease of measured RTT as an indication of potential extra bandwidth [12], eliminating the need for blind, periodic probing. Nevertheless, the coordination among a group of receivers in rate adaptation remains an issue.

Another problem of RLM is its poor rate adaptation performance in best-effort networks in which packet losses can occur even when transmission rate is well below the bottleneck rate. As RLM uses packet loss as an indication of congestion, heavy random packet drop will prevent optimal transmission rate from being attained. Moreover, under bursty background traffic, the available bandwidth can be time-varying and fluctuates rapidly. It is difficult for RLM to have a fast enough response for rate adjustment. The simulations in [10] have studied the performance of RLM. The results show that RLM operates far from the optimal level under bursty traffic.

An alternative scheme, Layered Video Multicast with Retransmission (LVMR) was proposed later to improve RLM [5] [6]. With this scheme, retransmission of lost packets is introduced. However, the issue of slow response in adaptation to rapid traffic condition changes remains.

## 2.2 New Features in IPv6

Internet Engineering Task Force has recently drafted a new version of Internet protocol known as IPv6 [7]. Apart from increasing the address space from 32 bits to 128 bits, several new features have been added in the new protocol to support real-time traffic and multicast. First of all, a range of IPv6 addresses is reserved for multicast purposes. Each IPv6 multicast address consists of a *Flag* field, a *Scope* field and a *Group* identifier. The flag field indicates whether the group is permanent or transient. The scope field limits the scope of the multicast group. The group identifier specifies the multicast group. In addition, a new *Traffic Class* field is added in the IPv6 packet header to provide QoS support. We now discuss the use of this class field.

IP packets arriving at a router are queued for processing. However, when the buffer of a router is full, the incoming packets will be discarded. The new class field allows a source to specify the desired priority/class of its packets. When a link is congested and the buffer overflows, the incoming and queued packets will be dropped from low to high class. Therefore, by assigning important real-time traffic to a higher class, certain degree of QoS guarantee can be attained.

## 3 Network-Driven Layered Multicast

Both RLM and LVMR attempt to adapt to network congestion and heterogeneity at the receiver side. It is true that receivers can successfully adjust to a more or less

optimal reception rate through these two approaches when the background traffic is smooth, but in practice, the bursty Internet traffic has limited their performance [10]. We propose to solve this problem using a Network-Driven Layered Multicast (NLM) scheme, which shifts the adaptation burden from receivers to the network. In our scheme, receivers are only responsible for handling packet loss recovery.

### 3.1 Video Layering Scheme

As other schemes mentioned in the previous section, our system makes use of layered video for multicast. It should be noted that our system is applicable to any layered encoding format. Here, we choose MPEG video as an example for illustration. In general, an MPEG video can be encoded into layers by scaling the video SNR, frame rate or a combination of both. For simplicity, we select a layering scheme similar to [5]; that is, we layer the video from the frame types. This simple layering technique requires a minimal additional effort from the encoder. A typical GOF pattern in a MPEG video sequence is  $I_1 B_2 B_3 P_4 B_5 B_6 P_7 B_8 B_9 P_{10} B_{11} B_{12} P_{13} B_{14} B_{15}$ . For layering, I-frames, which can be decoded independently, is assigned to the base layer. P-frames ( $P_4 P_7 P_{10} P_{13}$ ), which must be decoded from previous I-frames or P-frames, make up the first enhancement layer. B-frames, which should be decoded from both previous and subsequent I-frames or P-frames, constitute the second enhancement layer. By using this layering scheme, each successive subscribed layer can improve the video quality through an increase in frame rate. Finer layering is also possible by subdividing B and P-frames into more layers.

### 3.2 Rate Adaptation Scheme

The main difference between our scheme and RLM or LVMR is that our rate adaptation is totally handled by the network instead of receivers. To handle different priorities of IP packets, routers must be capable of processing the class field in the IPv6 header. Although nowadays most routers in the Internet are designed for IPv4, the classical IP, the existing routers will very likely be upgraded to become IPv6-compatible in the near future in order to support real-time traffic and the new address space. We also assume that the routers will be able to support a fair sharing of bandwidth among all the existing flows. That is, each flow will occupy only a certain proportion of the available bandwidth.

In NLM, rather than multicasting each video layer to a different IP address, we transmit all layers of the video source with a single multicast address, but each layer will be assigned with a different traffic class/priority. As each user must receive the base layer to acquire the minimal perceived quality, a higher priority is assigned to the base layer. For enhancement layers, priorities decrease up the layers since each additional layer only further refines the quality. In this way, the sender will transmit the video at a full rate to all receivers regardless of their network capacities.

It might seem at first glance that the multicast traffic under this approach will occupy all the bandwidth in low-speed links and prevent other flows from acquiring adequate bandwidth. However, with advanced routers, the available bandwidth in the links will be partitioned and reserved for different services. The multicast traffic will occupy only an allocated amount of bandwidth when a link is also used by other

services. Since each layer is multicast with a different priority, when the flow rate exceeds the allocated bandwidth portion, packets in the flow will be discarded by the routers starting from those of the lowest priority, which correspond to the highest enhancement layer. As a result, only those layers which the network capacity can sustain can traverse the links and reach receivers. The effect of this process is indeed similar to dropping a layer in RLM when congestion is detected at the receiver. Figure 2 depicts an example of NLM in a heterogeneous network.

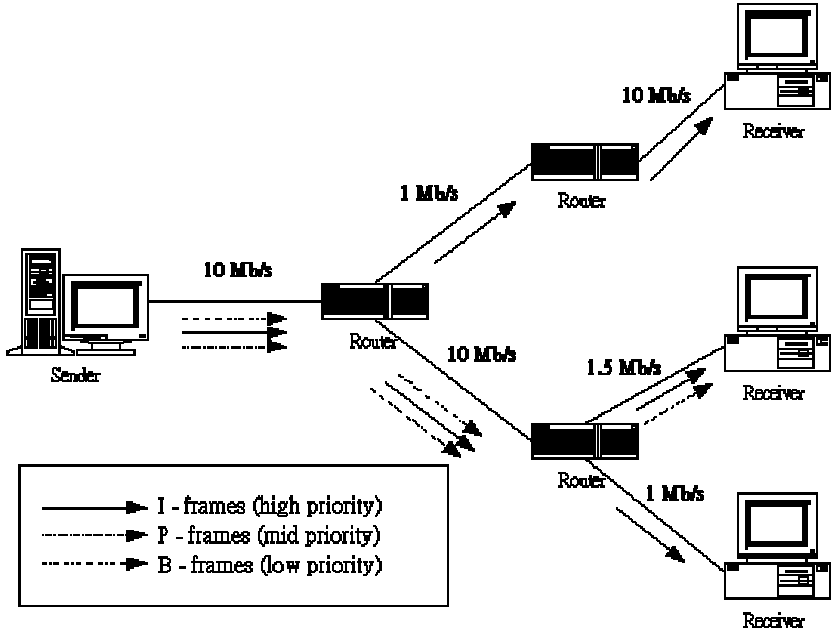


Fig. 2. An example of NLM

In practice, a user may not exactly receive a complete set of layers. For example, the network capacity may allow a user to receive only the base layer and part of an enhancement layer. In that case, we should decide whether to display the partially received layer since display of an incomplete enhancement layer can result in a poorer video quality than just playing the base layer. In NLM, we set a threshold  $\alpha$  on the proportion of layer received. If the proportion of layer received  $r$  is less than  $\alpha$ , the incomplete layer will be omitted in the presentation; otherwise it will be played with other lower layers. We will not suggest a value for  $\alpha$  here because its value depends on the particular video coding technique. If the video is vulnerable to data loss, then a threshold, say, as large as 90% - 95% should be set. In case a video can withstand some missing data without severe deterioration in perceived quality, a smaller value such as 70% - 80% may be assigned to  $\alpha$ . For the MPEG video in our example, the first enhancement P-frame layer should have a large  $\alpha$  since any missing data will be propagated to the subsequent P-frames and significantly degrade the perceived video quality. But the second enhancement B-frame layer can have a smaller threshold



because a B-frame will not be referenced by other frames for decoding. Any data loss in the layer will affect only the corresponding frame. Using a suitable error concealment technique, the impact of packet losses can be minimized.

During a video session, the traffic condition of the network may vary and hence a receiver should continuously monitor the proportion of a layer being received  $r$  in order to optimize the perceived video quality. Suppose at the beginning  $r < \alpha$  and the layer is not displayed (but is still being received). If now there is spare bandwidth, more packets of the layer are received, so  $r$  may then increase beyond  $\alpha$ . This will immediately trigger the receiver to start displaying the layer as well. Similarly, when network congestion occurs,  $r$  may drop below  $\alpha$ . Then the receiver should stop displaying the layer until available bandwidth increases again.

### 3.3 Retransmission-Based Error Recovery Scheme

Even when the network can support the delivery of different video layers by multicast, due to the best-effort nature of IP networks, packets can still be lost in transmission. Therefore, apart from the rate adaptation scheme discussed in the previous section, we further improve the performance of our multicast system by a retransmission-based error recovery scheme. We choose ARQ rather than FEC for error recovery because ARQ has more benefits than FEC as described in Section 1. Important issues such as retransmission requests, local recovery and priority of retransmitted packets are considered in our scheme.

#### a. Triggering of Retransmission Request

The function of retransmission is to recover a randomly lost packet. For packet loss due to heavy network congestion, a packet should not be retransmitted as this would only lead to congestion collapse. Thus, in our system, retransmission requests will only be triggered by packet losses in the displayed layers (i.e., layers with  $r > \alpha$ ) but not the undisplayed layers, since the losses in these layers are mainly caused by network overloading.

To detect packet loss, the popular approach is to check for gaps in received packet sequence numbers. Once a gap is discovered in the sequence space, it indicates that the expected in-sequence packet is probably lost. If packet losses are independent and not consecutive, this indeed offers a fast method to detect packet loss. However, it is recently found that bursty losses occur frequently in the Internet [8], which means that with this approach, packet loss may be detected at receiver only when a new packet arrives after several lost packets' inter-arrival times. To improve the response, we use both packet disorder and a time-out mechanism to trigger retransmission requests. Our mechanism works as follow: packets of each layer will have their own set of sequence numbers. When the packets in a layer are received out of sequence, and if the layer is a displayed layer, retransmission requests will be issued immediately. In addition, we also set a time-out instant for each coming packet. MPEG video has a variable encoding rate and it is difficult to predict the time of arrival for each packet, but by layering the video by frame types in our example, each layer can have a more regular bit rate. Moreover, we can use traffic shaping technique to smooth the video traffic so that packets in each layer can be transmitted more regularly. Thus, for each layer, it is possible to estimate the arrival time of packets and set the time-out instant

accordingly. We apply the streaming protocol proposed in [11]. Assume for layer  $i$ , the first packet (packet 0) is received at time  $t_0^i$ . If the mean inter-arrival time of packets in this layer is  $R^i$ , then the expected arrival time for packet  $j$  will be

$$e_j^i = T_j^i - \tilde{T}_j^i . \quad (1)$$

If the packet actually arrives at time  $T_j^i$ , then the error of expected arrival time is given by

$$\tilde{T}_j^i = t_0^i + jR^i; j \geq 0 . \quad (2)$$

In order to adapt to delay jitter and clock asynchronization between the sender and receiver for estimation of the packet arrival time,  $e_j^i$  is smoothed by an equation similar to the one used in TCP for estimating RTT:

$$e_{j\text{smooth}}^i = (1 - \lambda)e_{j-1\text{smooth}}^i + \lambda e_j^i . \quad (3)$$

where  $\lambda$  is a smoothing constant between 0 and 1. Therefore the estimated arrival time for packet  $j$  in layer  $i$  is

$$A_j^i = \tilde{T}_j^i + e_{j-1\text{smooth}}^i . \quad (4)$$

In our scheme, the time-out instant for packet  $j$  is set to be  $A_j^i$  plus a safety margin which is related to the deviation of estimated packet arrival time from the actual arrival time. The deviation for packet  $j$  is given by

$$\sigma_j^i = |T_j^i - A_j^i| . \quad (5)$$

We can use a smoothed value of  $\sigma_j^i$  to set the time-out instant. Specifically,

$$\sigma_{j\text{smooth}}^i = (1 - \lambda)\sigma_{j-1\text{smooth}}^i + \lambda\sigma_j^i . \quad (6)$$

Therefore the time-out instant for packet  $j+1$  in layer  $i$  is

$$TO_{j+1}^i = A_{j+1}^i + \beta\sigma_{j\text{smooth}}^i . \quad (7)$$

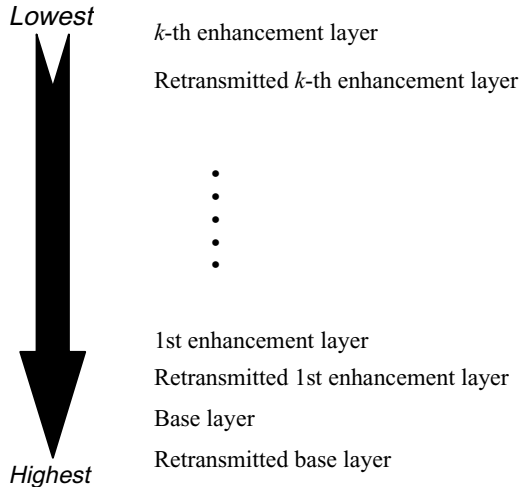
for some constant  $\beta > 1$ .  $\beta$  controls the sensitivity of the time-out triggering. From TCP recommendations, a typical value of 4 can be used. Accordingly, if a packet is dropped in the network and a time-out occurs or packet disorder is detected, a retransmission request will be issued. However, rather than multicasting the request back to sender for retransmission, the local recovery technique described in [4] is deployed in order to reduce the RTT of retransmitted packets.

### b. Local Recovery

In NLM, designated receivers (DRs) are used to retransmit lost packets in each subnet. A DR can be considered as a representative for a subnet. Its role is to reduce the load of network/source in retransmissions and the latency of retransmitted packets. When a member in the subnet detect a packet loss by the mechanism in (a), it will first multicast a NACK to its subnet so that the DR and all other receivers can get the request message. The NACK scope can be confined to the subnet only by setting the scope field in IPv6 multicast address. In this way, when other receivers get the NACK, if they also undergo the same packet loss, they need not issue the same request again and so prevent NACK implosion. In response to the request, the DR will then multicast the requested packet to its subnet so that the same loss at different receivers can be recovered simultaneously by a single packet. But if the DR misses the packet also, it will in turn multicast the retransmission request back to original source for recovery, and the source will multicast the missing packet to the whole multicast group.

### c. Priority Assignment for Different Packets Types

As for the first transmission of packets, retransmitted packets in different layers from the source should also have different priorities. Nevertheless, as retransmitted packets should reach the receivers as soon as possible before their play-out times, they should have an even higher priority than original packets in the same layer. Hence, we can assign priorities to the packets as follow: First, each layer can have a different priority. Then for each layer, we further subdivide the priorities according to whether the packet is an original or retransmitted one from the source. The resulting priority assignment for different packet types is shown in figure 3. Note that if a layer is only partially received (undisplayed layer), the lost packets will not be recovered by retransmission.



**Fig. 3.** Priority Assignment for Different Packet Types

## 4 Performance Study of NLM

NLM is more robust at rate adaptation and error recovery than RLM. We provide our rationale and give a brief performance analysis of NLM here. A detailed study by simulations will be left as future work.

### 4.1 Rate Control

One obvious benefit of NLM over RLM is that only one IP multicast address is required to transmit all the layers in multicasting. In RLM, each layer requires one address for multicast. Therefore, NLM has effectively reduced the IP address consumption in rate control, especially when the number of layers is large. However, the benefits of NLM are not limited to this. In receiver-driven approaches, rate adaptation requires the interaction between different receivers for coordination of the probing experiments (known as shared learning) [2]. This process involves a lot of overhead and complicated control messages. Although LVMR attempts to suppress the control traffic, it requires the installation of Intermediate Agents which will consume extra resources. A particular bad situation can arise in receiver-driven approaches is when one receiver misbehaves by subscribing more layers than available bandwidth in the subnet, all other users in the same subnet will be then affected and undergo network congestion. In NLM, this does not occur.

Another improvement of NLM over receiver-driven approaches is the way to detect spare bandwidth. In RLM or LVMR, periodical probing is used to check for any extra bandwidth. This method is inefficient because the reception quality at receivers is affected if no spare bandwidth is found during the probing experiment. Also, it is difficult to estimate the optimal probing period so as to minimize the frequency of failed experiments but can still provide a quick response to spare bandwidth. However, in NLM, when there is any unoccupied bandwidth, low priority packets will immediately be able to flow through the routers and the number of layers received will increase automatically without any effort from the receiver. This greatly reduces the response time in rate adjustment. These are also the reasons why we do not just combine RLM with IPv6 priority dropping feature. Although this combination can surely outperform RLM as base layer packets can have more protection against loss to ensure a basic video quality, the problem of complicated interaction between receivers and inefficient spare bandwidth detection will continue to exist in such an approach. Nevertheless, there is one drawback in our bandwidth detection approach. If the free bandwidth found is not located in the bottleneck link, then any increase of number of layers in a flow will not help in improving the video quality at receiver side as the additional layers will eventually be discarded in the bottleneck link.

### 4.2 Error Recovery

Apart from rate control, NLM also outperforms the receiver-driven approaches in error recovery. First of all, we use a fast retransmission technique to detect packet loss and send NACK. Suppose the average inter-arrival time for packets in layer  $i$  is  $t_i$  ms.

If the loss burst length is  $k$ , then using error recovery by gap detection only, it takes  $(k+1)t^i$  ms to trigger a retransmission request. However, with our time-out mechanism, assume we have an accurate estimation of packet arrival time, then it requires approximately  $t^i + \beta\sigma^i$  ms for the triggering, where  $\sigma^i$  is the mean deviation of packet arrival time in layer  $i$  and  $\beta$  is a constant. For the time-out mechanism to be effective, we require

$$t^i + \beta\sigma^i < (k+1)t^i. \quad (8)$$

$$\beta\sigma^i < kt^i. \quad (9)$$

Assume  $\beta$  is equal to 4 and the burst length  $k$  is 2, we have

$$2\sigma^i < t^i. \quad (10)$$

For a typical Internet video running at 128kbps with packet size of 1000bytes,  $t^i$  is larger than 60 ms and increases up the layers. And under normal Internet traffic conditions, the mean deviation of packet arrival time should lie between 10 - 20 ms. Hence, the inequality is valid in general. As a loss burst length of 2 packets is common in today's Internet traffic, this retransmission scheme works well together with the gap detection scheme. Moreover, the deployment of local recovery technique in our system further reduces the latency of retransmitted packets, which means that lost packets can have a higher chance of recovery. Even if the losses cannot be recovered by local DRs, the high priority of retransmitted packets from the source still offers a reliable retransmission channel for recovery.

## 5 Conclusions

We have proposed a scheme for Network-Driven Layered Multicast of video over the Internet. Our system made use of traffic class in IPv6 to overcome network heterogeneity and packet losses simultaneously. To perform rate adaptation, each video layer is assigned with a different priority. The priorities are assigned in such a way that only base video signal can flow through the network in times of congestion. We also suggested a retransmission-based error recovery scheme to deal with packet losses. Our scheme offers a faster packet loss detection technique by an integrated approach and a more effective retransmission method by the local recovery technique. A novelty with our error-recovery scheme is that we set a threshold on the proportion of data received in a layer to decide whether to perform loss recovery in that layer and display it. This cuts down unnecessary retransmissions of data of layers that will not be displayed and prevent exacerbating network congestion further.

## References

1. E. Amire, S. McCanne, and H. Zhang, "An Application-level Video Gateway," in *Proceedings of ACM Multimedia '95* (San Francisco, CA, Nov. 1995), ACM, pp. 255-265

2. S. McCanne, V. Jacobson, and M. Vetterli, "Receiver-driven Layered Multicast," in *Proceedings of SIGCOMM '96* (Stanford, CA, Aug. 1996), ACM, pp. 117-130
3. S. Pejhan, M. Schwartz, and D. Anastassiou, "Error Control Using Retransmission Schemes in Multicast Transport Protocols for Real-Time Media," in *IEEE/ACM Transactions on Networking*, Vol.4, No.3, Pages 413-427, June 1996.
4. S. Floyd, V. Jacobson, C.-G. Liu, S. McCanne, and L. Zhang, "A Reliable Multicast Framework for Light-Weight Sessions and Application Level Framing," in *Proceedings of SIGCOMM '95* (Boston, MA, Sept. 1995), ACM, pp.342-256.
5. X. Li, S. Paul, P. Pancha, and M.H. Ammar, "Layered Video Multicast with Retransmission (LVMR): Evaluation of Error Recovery Schemes," in *Proceedings of NOSSDAV '97*.
6. X. Li., S. Paul, and M.H. Ammar, "Layered Video Multicast with Retransmission (LVMR): Evaluation of Hierarchical Rate Control," in *Proceedings of IEEE INFOCOM '98*.
7. S. Deering and R. Hinden, "Internet Protocol, Version 6 (IPv6) Specification," *RFC2460*, Dec. 1998.
8. V. Paxson, "End-to-End Internet Packet Dynamics," in *Proceedings of SIGCOMM '97*.
9. T. Tuletli and J.-C. Bolot, "Issues with Multicast Video Distribution in Heterogeneous Packet Networks," in *Proceedings of Sixth International Workshop on Packet video* (Portland, OR, Sept. 1994).
10. S. Bajaj, L. Breslau and S. Shenker, "Uniform versus Priority Dropping for Layered Video," in *Proceedings of SIGCOMM '98*.
11. P. Wu and S.C. Liew, "A Streaming-Protocol Retransmission Scheme without Client-Server Clock Synchronization," in *IEEE Communications Letters*, Vol.3, No.7, July 1999.
12. C.W. Fung and S.C. Liew, "End-to-End Frame-Rate Adaptive Streaming of Video Data," in *Proceedings of ICMCS '99*.

# Analysis of the Requirements for ATM Multicasting Based on Per-PDU ID Assignment<sup>1</sup>

Josep Mangues-Bafalluy and Jordi Domingo-Pascual

Advanced Broadband Communications Center, Computer Architecture Department.

Polytechnic University of Catalunya.

Campus Nord. Mòdul D6. Sala 008.

Jordi Girona 1-3 – 08034 Barcelona (Spain)

{jmangues, jordi.domingo}@ac.upc.es

<http://www.ccaba.upc.es>

**Abstract.** The main issue to solve for ATM multicast forwarding is the cell-interleaving problem using AAL5. In strategies using a multiplexing ID, there are two main options: to assign an ID per source or per PDU. This paper argues for the convenience of using PDU ID because it allows the share of IDs between all the senders, thus reducing the ID consumption and management complexity. Compound VC (CVC) goes a step beyond in proposing a variable-length ID to adapt to the group size and to reduce the overhead introduced by the multiplexing ID. The size of this ID is negotiated at connection establishment according to the traffic and group characteristics. This paper provides some hints towards obtaining some dimensioning rules for the number of IDs based on its dependencies on traffic and group parameters.

## 1 Introduction

When dealing with ATM multicast forwarding, though ATM is cell-based, one expects the information in higher layers to be generated as packets. AAL5 seems to have been widely accepted and used for transmitting most data and multimedia communications over ATM. But AAL5 has no fields in its SAR-PDU allowing the multiplexing of cells belonging to different CPCS-PDUs. Therefore, the cell-interleaving problem must be solved in order to allow the receivers to reassemble the original packet when merging occurs.

One way to offer multicasting over ATM is IP multicasting over ATM [1], though it doesn't fully exploit ATM possibilities. A more efficient option is to provide multicasting mechanisms at the ATM level, that is, Native ATM Multicasting mechanisms. This term refers to those mechanisms implemented at the switches to allow the correct ATM level forwarding of the information being interchanged by the members of a group. There is no AAL5 CPCS-PDU reassembly inside the network.

A classification of these mechanisms can be found in [2], though it is slightly modified in [3] according to our conception of the mechanisms and to introduce our

---

<sup>1</sup>This work has been funded by the Spanish Ministry of Education (CICYT) under grant number TEL99-1117-C03-03

proposal. Thus, table 1 presents the classification as we think it should be with some more new mechanisms than in [2].

Techniques belonging to the first type solve the cell-interleaving problem by avoiding cells from different packets to be interleaved. VC Merging techniques ([4], [5]) reassemble all the cells of each PDU in separate buffers and forward them without mixing cells of different buffers (or PDUs). SMART [6] uses a token passing scheme to allow just one sender to put data in the multicast tree at any instant. In the second type , the VPI identifies the connection and the VCI is used as the multiplexing ID (identifying the source [7] or the PDU[8]). Compound VC switching [3] also uses multiplexing IDs per packet, but it allows the negotiation of its length (see Section 3). And the last type allows multiplexing inside the same VC either by adding overhead in the transmitted data ([9],[10]) or by using the GFC field in the header of the ATM cell [11].

**Table 1.** Classification of Native ATM Multicasting mechanisms

Avoid Cell-interleaving		VP switching		Compound VC switching	Allow Multiplexing inside a VC	
VC Merging (Buffering)	Token control	Source ID	Packet ID		Added overhead	GFC
SEAM		Standard VP switching			SPAM	Subchannel (WUGS)
VC Merge (MPLS)	SMART	VP-VC switching	DIDA	CVC	CRAM	

Those strategies using multiplexing IDs may be classified in two main groups: Source IDs and Packet Data Unit (PDU) IDs. In strategies using Source ID, the ID is related to the source that transmitted the packet. On the other hand, PDU ID strategies assign an ID (independent of the source) to each packet. The next section explains both philosophies in further detail.

This paper argues for the convenience of using PDU IDs instead of Source IDs for multiplexing, due to the efficiency in terms of ID consumption. The advantages of variable-length PDU IDs and the flexibility they allow in group size by providing minimum overhead are also discussed. The next step is to determine the correct size of the PDU ID according to the traffic characteristics and the group size. For that purpose, we carried out some simulations. Thus, the goal of this paper is to find a methodology to study the dependence on traffic and group characteristics of the PDU losses due to running out of identifiers at a given switch.

A discussion on Source ID and PDU ID advantages and drawbacks is presented in the next section. Following that, there is a brief description of the Compound VC mechanism, as it is the mechanism for which calculating the size of the PDU ID makes sense. Determining the size of the ID is the issue of the following section.



Section 5 discusses the results of our simulations. Finally, conclusions and future work are presented in the last section.

## 2 Source ID and PDU ID

The classification presented in table 1 is based on the way each strategy employs to solve the cell-interleaving problem. Except for the mechanisms labeled as ‘Avoid Cell-Interleaving’, which use buffers or a token, all the rest use some kind of multiplexing ID (muxID) to deal with this problem. The purpose of the muxID is to identify each particular cell so as to make the end-system able to correctly reassemble all the PDUs it receives.

Multiplexing IDs allow cells belonging to different PDUs to be interleaved, and thus the traffic characteristics of all the sources in the group are respected, making these strategies suitable for multimedia communications. However, in VC merging strategies, the buffer requirements and the increase in CDV and burstiness due to buffering limits its application to multimedia communications. The connection management complexity of SMART may also limit its application to multimedia communications. Therefore, time-constrained traffic may be more suitably served by allowing multiplexing of cells belonging to different PDUs. The price paid by muxID strategies is the extra overhead to carry the ID, which adds to the intrinsic ATM overhead.

In Source ID mechanisms, the ID is related to the source that transmitted the packet. Therefore, there is a binding between the source and the ID at each switch. This binding must be unique at each switch so as to avoid ID collision at merge points. The ID collision problem may be solved either by globally assigning IDs for the group or by locally remapping the IDs at each switch [7]. The management required in the former option may limit its scalability. On the other hand, local remapping maintains a list of free IDs at each switch, where a local mapping of IDs is carried out.

Source ID mechanisms usually overdimension the size of the ID so as to solve the worst case in which there could be a lot of senders (usually up to  $2^{15}$  or  $2^{16}$ ). However, not all groups will have such a huge number of senders, and most overhead will be unused, e.g. in the local area.

PDU ID strategies assign an ID to each packet, and this assignment is independent of the source this packet came from. A new incoming PDU to the switch is assigned an ID from a pool of free IDs. Thus, packets coming from all the sources in the group share the identifiers. In this way, ID consumption is smaller than with Source ID. However, the solutions proposed up to now also use fixed size identifiers except in one case, Compound VC [3]. DIDA uses a 16-bit field, which is also overdimensioned, even more than in the Source ID case, because these IDs are shared by all the senders. GFC, on the other hand, uses small IDs (the 4 bits of the GFC field), which may be insufficient for bigger groups. In this case, more than one such GFC-connection should be used and the group management is then increased.

The exception comes from Compound VC (CVC), which allows flexible ID size negotiation at connection establishment so as to adapt to the ID consumption required

by each group. In this way, the overhead is minimized for two reasons: the PDU ID philosophy and the ID size negotiation. The next section briefly explains how CVC works.

### 3 Compound VC (CVC)

The main goal of the CVC mechanism [3] is to solve some of the problems of the mechanisms that have been commented above, namely buffer requirements, alteration of traffic characteristics, overhead, and flexibility.

Like in DIDA, there is a multiplexing ID per PDU that is carried by each cell belonging to that packet. CVC is based on local remapping of multiplexing IDs at the switches. Thus, no global ID assignment mechanism is needed, which would limit its potential deployment in wider environments than the local area.

VP switching techniques use the VPI to identify the group. This limits scalability due to VPI exhaustion, and also the exclusive use of the VPI by the operator. The solution CVC proposes is to treat a group of VCs as a whole when switching cells. The number of VCs in this group could be a power of 2 between one and  $2^{16}$ . Thus, by providing this flexibility, scalability is increased. But scalability must not limit the potential application of a mechanism to multimedia, which is growing in importance. Multimedia traffic imposes stringent QoS constraints that make mechanisms that interleave cells more suitable due to traffic and group interaction constraints.

In CVC, the size of the PDU ID is negotiated at connection establishment depending on the group and traffic characteristics. The number of bits used as multiplexing ID may be determined by means of a mask. As far as the architecture is concerned, CVC requires a dynamic updating of the tables at the switch each time the first cell of a new PDU arrives and each time the last cell of a PDU arrives. Therefore, a new column in the switching table is required to consider a mask, but table size is not globally increased as there is just one entry for the whole group. The mask determines the portion of the VCI that will identify the group of VCs and the portion that will contain the multiplexing IDs (see fig. 1). CVC masks allow a lot of smaller groups with diverse sizes where there was just one group connection inside a VPI.

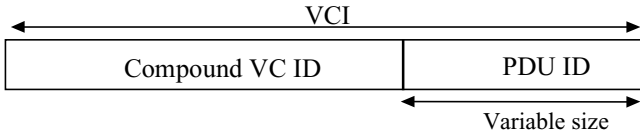


Fig. 1. VCI field in Compound VC

### 4 Determining the Number of Identifiers

The two preceding sections served to argue for the convenience of having PDU IDs of variable length to solve the cell-interleaving problem. CVC allows such philosophy. Therefore, one of the main problems to solve for CVC is the dimensioning of such variable-length ID.

The dimension of the PDU ID at a given switch through which a CVC connection passes should be chosen according to the traffic characteristics and the group size. The characteristics of the aggregated traffic observed by a switch are related with the traffic characteristics of each particular sender, that is why in this paper we try to determine the dependence of the required number of IDs as a function of the parameters of one source. This makes sense because we will study a homogeneous environment, i.e. all the senders transmit traffic with the same characteristics. The traffic parameters considered in this initial study are the PCR, the mean traffic and the length of the PDU.

With respect to group characteristics we see that passive members of the group, i.e. hosts that only act as receivers, do not have any effect on this traffic. As a consequence, the group size is characterized by the number of senders (N) in the group, when the purpose is to dimension the ID.

The goal of this paper is to find a methodology to study the dependence on traffic and group characteristics of the PDU losses produced due to running out of identifiers at a given switch. The procedure we will follow will start by calculating the histogram representing the frequency of the number of slots in which a given number of simultaneous PDUs is being transmitted through a given output port, i.e. the discrete probability density function of the number of simultaneous PDUs. Next, a graph for the PDU loss probability due to running out of identifiers is calculated, i.e. the graph will represent the probability of loss of an arriving PDU as a function of the number of IDs (nID) used. This graph is obtained from the previous one by adding from nID up to N-1 simultaneous PDUs. This sum is carried out for nID values from 0 to N. The probability that a PDU passes through a switch as a function of nID may also be obtained from the initial graph. In this case, we add the values ranging from 0 to nID-1 simultaneous PDUs.

As one of the goals of the paper is to obtain a value for nID as a function of traffic and group characteristics, our study will focus on the PDU loss probability graph. The idea is to find some dimensioning rules that relate such probability with the parameters being considered. Such rules would allow dimensioning the PDU ID at CVC connection establishment given a PDU loss probability acceptable for the user.

An analytical expression for finding the probability that a burst of the GFC mechanism is lost is presented in [11].

$$\sum_{i=h}^{n-1} \binom{n-1}{i} p^i (1-p)^{(n-1)-i} \quad (1)$$

This equation depends on the number of sources (n), the number of subchannels (h), the average of busy sources (m), and p is the probability that any given source is transmitting a burst ( $p=m/n$ ). When applied to CVC, a burst is taken to be a PDU.

However, parameter m is too generic to be useful at connection establishment when trying to determine the number of required IDs. Other more easily obtained parameters, which could be directly related to the source, should be used for this purpose.

## 5 Simulation Environment

The simulated scenario consists of some bursty sources sending traffic to the same output port of a switch. They are characterized by their average cell rate ( $R$ ), their peak cell rate ( $P$ ), and the number of cells per burst ( $B$ ). The sources are homogeneous, i.e. they are all modeled by means of the same statistics with the same parameters. A MMDP (Markov Modulated Deterministic Process) with one active state and one silence state is used to model each source, which is a particular case of ON-OFF source (figure 2). For this model, the sojourn time at both states follows a geometric distribution. In the OFF state, the source does not send any cells. In the ON state, it sends cells at its peak cell rate ( $P$ ).

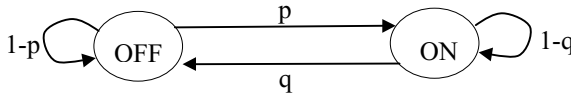


Fig. 2. ON-OFF source model

From the analysis of this Markov chain we may obtain that the probabilities of being at the ON and OFF states are:

$$P_{ON} = \frac{p}{p+q} \quad P_{OFF} = \frac{q}{p+q} \quad (2)$$

and the transition probabilities are related to the source parameters as follows [12]:

$$p = \frac{1}{B(b-1)} \quad q = \frac{1}{B} \quad (3)$$

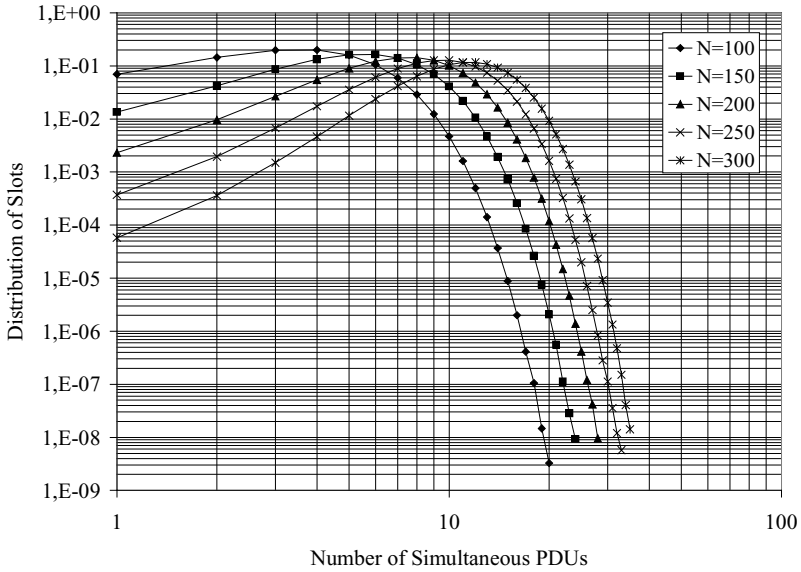
where  $b=P/R$  is the burstiness of the source.

Each source sends just one PDU at any given time. In our simulator, the output queue is modeled as a counter of the number of simultaneous PDUs for each slot. During all the simulations, the number of sources is varied depending on the mean traffic introduced to the switch so as to assure that the losses only occur due to running out of identifiers and not due to overload. This assumption is possible if we consider that there is enough buffer space at the switch to absorb the burstiness due to the aggregation of sources. The simulated time is  $10^{10}$   $\mu$ s.

The reference source is characterized by the following parameters. At peak cell rate (PCR), the source transmits one cell out of fifteen, i.e. for an STM-1 link the PCR is 10 Mbps. The mean traffic introduced by each source is 0.5 Mbps. The number of sources ranges from 100 to 300 to study the behavior of the mechanism for mid and high loads without reaching instability. PDUs are composed of an average of 5 cells and the sojourn time at ON state follows a geometric distribution. The OFF state also follows a geometric distribution with a mean of 1425 cells, which was chosen to obtain an average of 0.5Mbps per source. Any variation with respect to these parameters will be noted when presenting the results.

## 6 Results

The distribution of the number of simultaneous PDUs for a given reference scenario is presented in figure 3. Logarithmic scale has been chosen for both axis to provide a further detail for the range of values of interest, i.e. between 16 (4 bits) and 128 (7 bits) identifiers. We focus on these values because few bits in the ID provide low losses due to running out of ID (see 4). Statistically non-significant values have been removed from the figure.



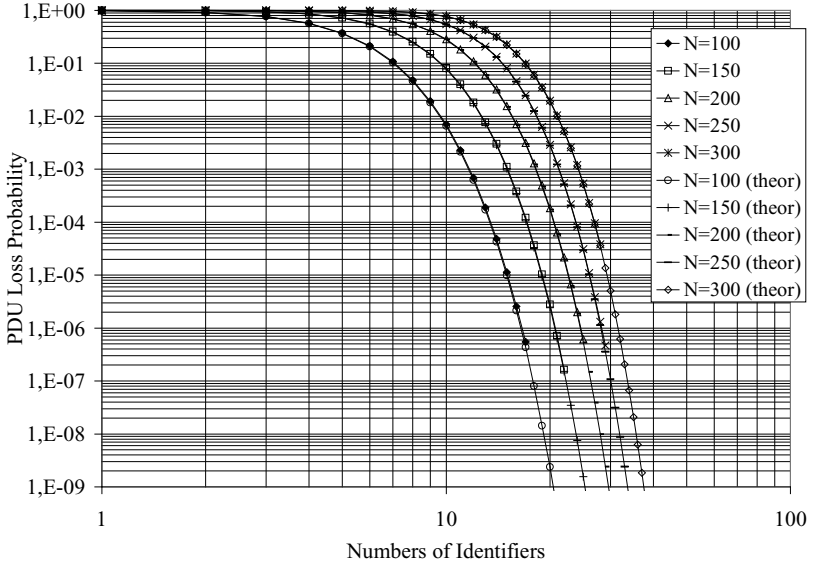
**Fig. 3.** Distribution of the number of simultaneous PDUs at the output port of a switch where merging occurs. The reference scenario is average=0.5Mbps per source, 5 cells per PDU, and PCR=150Mbps. Each curve corresponds to a different number of sources ( $N$ ).

Figure 4 represents the probability that an arriving PDU does not find a free ID at the switch. Each curve corresponds to a different number of sources, and thus, different average loads at the switch, ranging from 50 Mbps for  $N=100$  to 150Mbps for  $N=300$ . Only values that presented small 95% confidence intervals are represented.

A comparison with the analytical expression commented above (equation 1) is also presented. The goal of the comparison is to study the goodness of the fit between the results obtained through simulation and the theoretical expression for the simulated range of values. The PDU loss curves obtained by applying the analytical expression are labeled as  $N$  (*theor*) in figure 4. The parameters appearing in the expression were mapped to parameters in our simulation to obtain such curves. A burst is taken to be a PDU in our simulation,  $n$  corresponds to the number of sources ( $N$  in the figures),  $m$

corresponds to the average number of simultaneous PDUs sent by the sources to the same output port, and  $h$  is the number of IDs. From the analysis of the expression it follows that each of the terms that are summed corresponds to the fraction of slots in which there are a given number of bursts being transmitted. In our case a burst corresponds to a PDU, therefore, figure 3 represents each of the terms being summed. And the sum from  $h$  up to  $n-1$  is exactly how we obtain the graph of the PDU loss probability. It can be observed that the results of the simulation and those of the expression coincide for those values with small 95% confidence interval.

In a similar way, from the sum of the rest of the values, i.e. from 0 up to  $h-1$ , we obtain the probability that an arriving PDU to the switch is correctly multiplexed and forwarded through the output port because it was able to allocate a free multiplexing ID. These results give us an idea of the throughput of PDUs. These curves are not presented because those of figure 4 provide the complementary information. Just remark that, for this scenario, with 16 identifiers, PDU throughputs near 90% are obtained for the highest load case ( $N=300$ ).

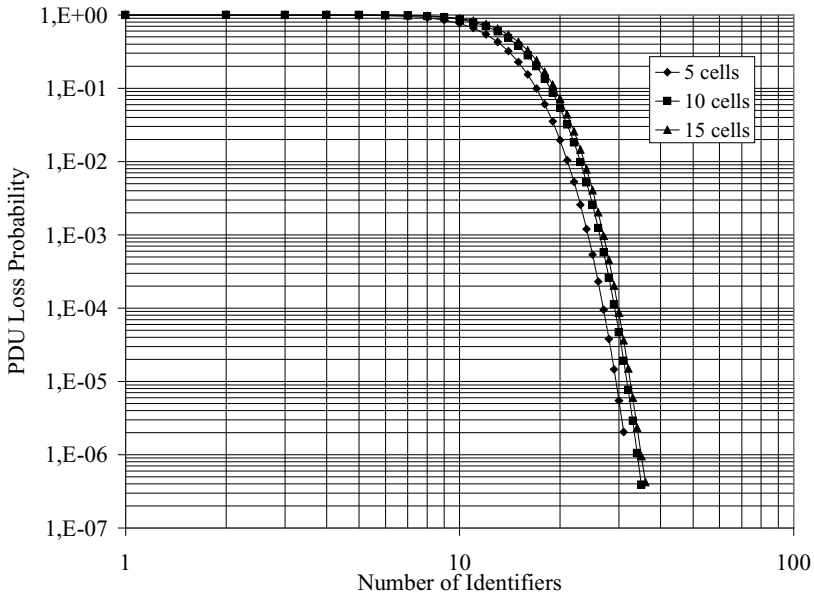


**Fig. 4.** PDU Loss Probability due to running out of identifiers. Reference scenario is: average per source = 0.5Mbps, mean PDU length = 5 cells, and PCR = 10 Mbps.

The main conclusion that may be drawn from these results is that with a few bits (between 4 and 6) and by using the PDU ID strategy, low PDU loss probabilities may be obtained even with a high number of sources. This scenario, which represents a possible scenario in future group communications, shows the advantages of PDU ID over Source ID multiplexing. Source ID requires a number of IDs assigned to the group equal to the number of sources. As a consequence, the overhead introduced by the multiplexing ID in these latter strategies is much higher than that of CVC. CVC allows the negotiation of the ID size to adapt to the traffic and group requirements.

Another characteristic that can be observed in figure 4 is the linearity (when using the logarithmic representation) of the PDU loss probability in the range of values of interest for the IDs. Therefore, the curve may be divided into three main linear regions. The first one starting at low ID values would be flat, which would tell that the number of IDs is not enough for such kind of traffic and group characteristics, as the PDU loss probability is 1. The second region, the one whose characterization is our main concern, would be a line going from the number of IDs where the curve starts to bend up to a number of IDs equal to  $N-1$ . The third region is characterized by a vertical line starting at  $nID=N$ , meaning that it is nonsense to use more IDs than sources in the group, because no losses occur due to running out of identifiers when  $nID \geq N$ .

Of course, such an approach is a rough approximation of the actual curve, but this characterization would allow a CVC switch to obtain, by means of a simple expression, the number of required IDs as a function of group and traffic characteristics and accepted PDU loss probability during connection establishment. The dependence of the parameters of these lines as a function of traffic characteristics is left for further study.



**Fig. 5.** PDU Loss Probability comparison varying the mean PDU length. Average per source=0.5 Mbps, PCR=10 Mbps, and  $N=300$ .

Other simulations have been carried out with different traffic and group parameters. For instance, PDU loss probability results were obtained for a number of sources ranging from 500 to 1500 with an average traffic per source of 0.1 Mbps. This average value is obtained by varying the mean sojourn time at OFF state. The same

observations stated above apply for this new scenario. The simulation results are also compared with the analytical ones. Both curves coincided for statistically significant values. However, simulations with high mean average per source (e.g. 5 Mbps) showed a difference between both curves.

Finally, the comparison between the results presented in figure 4 (reference scenario) and those obtained for the scenario with average per source=0.1 Mbps showed an almost imperceptible variation in the curves when the aggregated average load was the same.

The rest of the curves presented in this paper provide some results to study the dependence of the PDU loss probability curves on each of the considered parameters. For instance, figure 5 presents a comparison of the curves obtained for  $N=300$  when varying the mean PDU length. The simulated values are 5, 10, and 15 cells per PDU, which correspond to reasonable mean values according to current Internet traffic, and in particular, to multimedia traffic. As it could be expected, the longer the PDU, the longer the IDs are occupied, and the more IDs are required. However, the variation between these curves is not very high. Thus, it may be observed that the curves show the same behavior (they all have the same shape). The only difference is a slight shift. Therefore, in the rough linear model we proposed to describe the behavior of these curves, it seems that the dependence of the equation of the line in the region of interest would be in the position of the point where the curve bends and not in the slope. Anyway, for reasonable mean values, such dependence would not be very strong.

Finally, figure 6 presents a comparison of the reference scenario described before with others in which one or two parameters are changed with respect to the reference one. These results were obtained for  $N=250$  sources, which produce an aggregated traffic of 125Mbps, as the average traffic per source is 0.5Mbps.

We first focus on the curves that just vary the PCR while maintaining the rest of the reference parameters. They are labeled as  $PCR=2Mbps$ ,  $avg=0.5$  (which corresponds to  $PCR=10Mbps$ ),  $PCR=30Mbps$ , and  $PCR=150Mbps$ . In this case, the range of possible values is wider than in the PDU case. The effect of varying the PCR is to modify the burstiness of each source. This is so because the average per source remains unchanged while the cells are being transmitted in longer or shorter ON periods.  $PCR=150Mbps$  corresponds to the most bursty sources, and  $PCR=2Mbps$  corresponds to the smoothest simulated traffic.

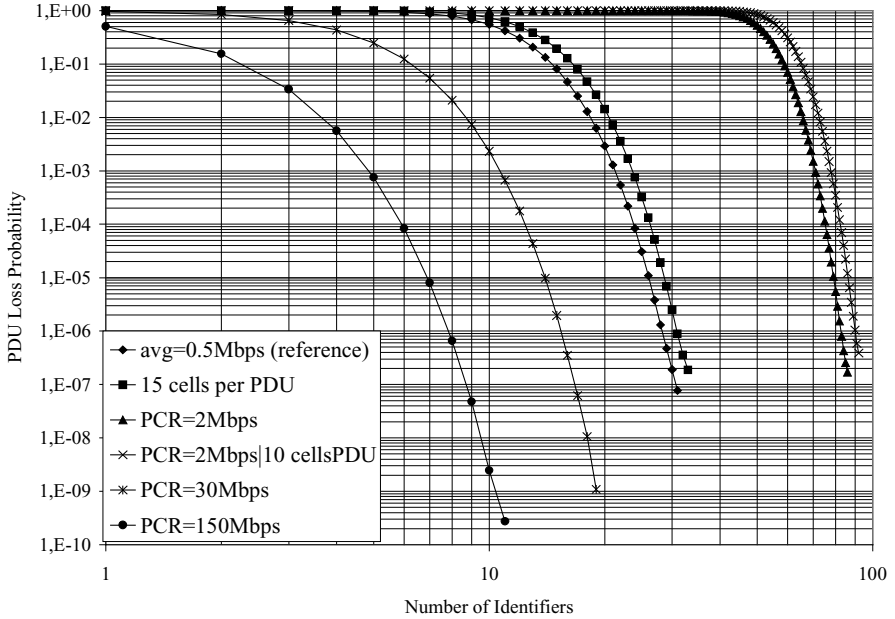
For instance, to obtain a PLP of  $1e-6$ , the number of bits required for the PDU ID is respectively 7 (128 IDs), 5 (32 IDs), 4 (16 IDs), and 3 (8 IDs) for the 2, 10, 30, and 150 Mbps cases. This is due to the relationship between the PCR and the burstiness of the traffic introduced by the source. That is, if we maintain the same average traffic per source and we vary the PCR, the same number of cells per PDU is sent, but at a higher speed. Therefore, the PDU lasts less and as a consequence it is using an ID during less time, making it possible for other sources to get that ID.

These results show a strong dependence of the shift of the curves with respect to the PCR. However, the slope of the line in the region of interest seems to show only a slight dependence on the PCR. These dependencies will be studied in future work.

The curve labeled as  $PCR=2Mbps/10\text{ cells per PDU}$  serves to confirm that the PLP curve is more sensitive to PCR variations, i.e. to the burstiness of the traffic, than to



the length of the PDUs. Its values are more similar to those of the curved labeled as  $PCR=2Mbps$  than to those obtained for the *15 cells per PDU* case.



**Fig. 6.** PDU Loss Probability comparison for different parameters

After examining the results for this scenario, and without aiming to generalize, we could give some hints for the dimensioning of the ID size for a given CVC connection. They are mostly based on the bursty behavior of the sources, which is the characteristic responsible for the most important variations in the PLP values obtained for a given aggregated average load.

We calculated the burstiness of the sources for which their PLP curves are represented in figure 6. The burstiness ( $b$ ) is 4 ( $=10Mbps/0.5Mbps$ ) for the sources whose curves are labeled as  $PCR=2Mbps$ , and  $PCR=2Mbps/10cellsPDU$ , 20 for  $avg=0.5$  and  $15 cellsPDU$ , 60 for  $PCR=30Mbps$ , and 300 for  $PCR=150Mbps$ . For a PLP of  $1e-6$ , the number of bits required is respectively, 7, 5, 4, and 3. Therefore, in this scenario, we could use these ID sizes for values of burstiness around those calculated. Further work is required to derive rules that apply to scenarios with different average per source. They should also take into account the aggregated load, whose importance has been noted in comparisons presented above.

The diversity in scenarios and requirements for different groups also shows the advantages of having flexible ID size negotiation, such as the one offered by CVC. For instance, in multimedia group communications more losses could be accepted for video than for audio, and different number of sources would require different number of IDs.

## 7 Conclusions and Future Work

The results presented in this paper confirm that the efficiency in terms of multiplexing ID overhead could be highly reduced with PDU ID strategies when compared to Source ID strategies.

The convenience of flexible ID size negotiation, such the one offered by the Compound VC (CVC) mechanism, can be deduced from the diversity in requirements and group characteristics.

The PDU loss probability in the range of values of interest, i.e. from 16 IDs (4 bits) to 128 IDs (7 bits), shows a linear trend in logarithmic representation. Finding the expression of this line as a function of the traffic and group characteristics would allow easy ID negotiation during connection establishment. The derivation of this expression is left for future work.

The results have shown that the expression proposed in [11] and the PLP curve obtained through simulation coincide in most cases. However, the application of such an expression at connection establishment is limited because the user does not know the mean number of simultaneous PDUs the connection will have. Parameters that are available at connection establishment should be used instead.

The results also showed that the PLP is more sensitive to PCR variations than to PDU length. Some simple hints for the dimensioning of the ID size were given for a reference scenario. Once the aggregated load is fixed, the most important parameter is the burstiness.

The price paid when deploying CVC is the extra complexity in the switches. Implementation issues of CVC are left as future work to determine whether the benefit of introducing CVC is higher than its cost.

Other traffic parameters should be considered in future simulations so as to be able to draw more general conclusions on the behavior of PDU loss probability. Other kinds of traffic types may also be introduced to simulate more real environments. And scenarios of heterogeneous traffic sources may also be of interest in our future work.

## References

- [1] Armitage, GJ 'IP multicasting over ATM Networks.' IEEE Journal on Selected Areas in Communications 15(3): 445-457, April 1997.
- [2] Baldi M, Bergamasco D, Gai S, and Malagrino D. 'A Comparison of ATM Stream Merging Techniques.' Proceedings of IFIP High Performance Networking (HPN'98): 212-227, Vienna, September 1998.
- [3] Mangues-Bafalluy J and Domingo-Pascual J. 'Compound VC Mechanism for Native Multicast in ATM Networks.' Proceedings of the 2<sup>nd</sup> International Conference on ATM (ICATM'99): 115-124, Colmar (France), June 1999.
- [4] Grossglauser M and Ramakrishnan KK. 'SEAM: Scalable and Efficient ATM Multicast.' Proc. of IEEE Infocom'97: 867-875, Kobe (Japan), April 1997.
- [5] Rosen EC, Viswanathan A, and Callon R. 'Multiprotocol Label Switching Architecture.' IETF Draft, draft-ietf-mpls-arch-05.txt, April 1999.
- [6] Gauthier E, Le Boudec J-Y, and Oeschlin P. 'SMART: A Many-to-Many Multicast Protocol for ATM.' IEEE Journal on Selected Areas in Communications 15(3): 458-472, April 1997.

- [7] : Venkateswaran R, Raghavendra CS, Chen X, and Kumar VP. 'Support for Multiway Communications in ATM Networks.' ATM Forum/97-0316, April 1997.
- [8] Calvignac J, Droz P, Baso C, and Dykeman D. 'Dynamic Identifier Assignment (DIDA) for Merged ATM Connections.' ATM Forum/97-0504, July 1997.
- [9] Komandur S and Mossé D. 'SPAM: A Data Forwarding Model for Multipoint-to-Multipoint Connection Support in ATM Networks.' Proc. of the 6th International Conference on Computer Communications and Networks (IC3N). Las Vegas, September 1997.
- [10] Komandur S, Crowcroft J, and Mossé D. 'CRAM: Cell Re-labeling At Merge Points for ATM Multicast.' Proc. of IEEE International Conference on ATM (ICATM'98), Colmar (France), June 1998.
- [11] Turner J. 'Extending ATM Networks for Efficient Reliable Multicast.' Proc. of Workshop on Communication and Architectural Support for Network-Based Parallel Computing, Springer Verlag, February 1997.
- [12] Bolla R, Davoli F, and Marchese M. 'Evaluation of a Cell Loss Rate Computation Method in ATM Multiplexers with Multiple Bursty Sources and Different Traffic Classes.' Proceedings of IEEE Globecom, pp. 437-441, 1996.

# Sensitivity of ABR Congestion Control Algorithms to Hurst Parameter Estimates

Sven A. M. Östring<sup>1</sup>, Harsha Sirisena<sup>1</sup>, and Irene Hudson<sup>2</sup>

<sup>1</sup> Department of Electrical & Electronic Engineering

<sup>2</sup> Department of Mathematics & Statistics

University of Canterbury, New Zealand.

{s.ostring,h.sirisena}@elec.canterbury.ac.nz

i.hudson@math.canterbury.ac.nz

**Abstract.** Optimal linear predictors can be utilised in ABR control algorithms for the management of self-similar network traffic. However, estimates of the Hurst parameter are required to generate these predictors, and uncertainty in these estimates results in a potential mismatch of the predictors to the traffic. When mismatched congestion control algorithms are used within the network, the impact on the network system is greater queue lengths at buffers and more significant cell losses. The sensitivity of these algorithms to the Hurst parameter estimate is investigated both analytically and using simulations. It is shown that an asymmetry in the sensitivity occurs in the region where the Hurst parameter is significantly underestimated.

## 1 Introduction

A significant amount of research in the area of teletraffic modeling has been focused on proving the self-similarity of network traffic [1,2]. Methods of accurately estimating the Hurst parameter, the index of self-similarity, is a key issue within this area, and these range from relatively simple methods, such as the variance-time plots and R/S statistic analysis [3], to more sophisticated techniques, such as Whittle's estimator and estimators based on the wavelet transform [4]. Having demonstrated that network traffic is self-similar, the subsequent step is to determine the impact of self-similarity on the network as a system. The performance of a queue is fundamentally different when the input process to the queue is self-similar [5], with the distribution of the queue length now being heavy-tailed. In the case of finite buffers, higher cell losses occur within the system and these losses decrease hyperbolically as the buffer size is increased, rather than the exponential decrease which occurs with Poisson processes [6].

These rather serious consequences of self-similar traffic has driven research, though a limited extent, to consider methods of recognising the characteristics of self-similarity within network resource management techniques [7,8,9].

Our research has focused on incorporating the characteristics of self-similarity into congestion control algorithms, and in particular rate-control mechanisms for the ABR service in ATM networks [10,11]. This work has demonstrated that the buffer memory requirements and cell losses can be reduced if the control algorithms are developed using the stochastic structure of the self-similar background traffic, and that adaptive algorithms based on on-line Hurst parameter estimators can be implemented which track non-stationarities which occur in the traffic.

An area which has not been addressed by any known research work is that of investigating the effect of poor knowledge of the actual Hurst parameter when resource management algorithms are designed specifically with self-similarity in mind. In this paper, this sensitivity is investigated more thoroughly, both through analysis of the relative change in variance of prediction errors resulting from the mismatched algorithm and through simulations, where data sets are tested with a range of algorithms developed from different Hurst parameter values.

## 2 Self-similarity of Network Traffic

### 2.1 Concepts and Models

There is significant statistical evidence that a wide range of classes of network traffic is self-similar in nature. This means that there is no natural length of the bursts in the traffic, and the traffic remains bursty over a range of time scales (hence the term “self-similar”). The Hurst parameter  $H$  is the index of self-similarity of a process, and a process is categorized as long-range dependent (LRD) when the parameter lies in the interval  $0.5 < H < 1.0$ . Long-range dependence means that the correlations within a process decrease hyperbolically rather than exponentially, so that the autocovariance function for LRD processes is non-summable. While the burstiness of LRD traffic can cause buffer overflows and losses, long-range dependence can be used to one’s advantage in terms of prediction [10,12]. These large correlations mean that there is significantly more information within previous states regarding the current state in LRD processes than in short-range dependent (SRD) processes, and more accurate predictions can be achieved from appropriately filtering stored measurements of the process in the past.

There are a number of well-known models used for processes which display self-similarity. Fractional Brownian motion is the canonical example of a self-similar process, and its incremental process (called fractional Gaussian noise—fGn), has the autocovariance function:

$$\begin{aligned} \gamma(k) &= \frac{\sigma_0^2}{2} (|k+1|^{2H} - 2|k|^{2H} + |k-1|^{2H}), \quad k \in Z \\ &\sim \sigma_0^2 H(2H-1)k^{2H-2} \text{ as } k \rightarrow \infty \end{aligned} \quad (1)$$

where the asymptotic behaviour of the autocovariance function shows that the process is long-range dependent. A self-similar process can be parsimoniously

represented by an fGn model, if the mean, variance and Hurst parameter of the process are known. Another important class of self-similar models is the fractional ARIMA family of models, which are a natural extension of the ARIMA( $p, d, q$ ) models where the differencing parameter  $d$  is allowed to assume fractional values.

## 2.2 Estimation of the Hurst Parameter

As stated in Section 2.1, the Hurst parameter plays a key role in characterizing the self-similarity of a process. Thus, the estimation of this parameter from a set of measurements is crucial in determining whether a process is self-similar, and has attracted a significant amount of research. Self-similarity manifests itself within data in three fundamental ways—slowly decaying variances as the data is aggregated, long-range dependence within the covariance structure and a spectral density which obeys a power-law with divergence near the origin. Methods of estimating the Hurst parameter are based on quantifying one of these behaviours, usually by estimating the slope of a graph based on some transformation of the data.

Two heuristic estimation techniques are variance-time analysis and the re-scaled adjusted range ( $R/S$ ) statistic analysis [3]. Variance-time analysis is based on the asymptotic relationship of the variance of sample averages  $X^{(m)}$  of non-overlapping blocks of data of size  $m$  from the process  $X$ , with the relationship given by:

$$\text{Var}(X^{(m)}) \sim cm^{-\beta}, \text{ as } m \rightarrow \infty \quad (2)$$

with  $0 < \beta < 1$  for LRD processes. The other well-known heuristic technique is  $R/S$  statistic analysis, where the  $R/S$  statistic exhibits the Hurst effect as described by the relationship:

$$\mathbf{E}[R(n)/S(n)] \sim cn^H, \text{ as } n \rightarrow \infty \quad (3)$$

with the parameter  $H$  typically about 0.7.

More refined methods of data analysis are based on maximum likelihood type estimates (MLE) and the use of periodograms which effectively transform the data into the frequency domain, to estimate the power-law behaviour near the origin. The well-known Whittle's estimator is an approximate MLE which is asymptotically normal and efficient. The Abry-Veitch (AV) estimator is a fast estimation technique based on the wavelet transform [4]. The wavelet domain is a natural framework from which to view self-similar processes due to the scaling behaviour which is common to both fields. The wavelet transform generates a set of detail coefficients  $d_{\mathbf{x}}(j, k)$  from a data set, and for a LRD process  $\mathbf{X}$ , the variance of the detail coefficients at each level  $j$  is given by the relationship:

$$\mathbf{E}[d_{\mathbf{x}}(j, \cdot)^2] = C \cdot 2^{j(2H-1)} \quad (4)$$

with  $C > 0$ . An important property of the AV estimator is that it can be reformulated to generate on-line estimates of the Hurst parameter [13]. This is

because it is based on the fast pyramidal filter, which was originally intended for on-line applications. These on-line estimates can be used within adaptive ABR control mechanisms [11].

### 3 Using Hurst Parameter Estimates in ABR Rate Control

The potential impact of self-similar traffic within networks, such as greater queue lengths at nodes and increased cell losses, makes it necessary to incorporate the characteristics of self-similarity into resource management algorithms. In particular, we have investigated using the properties of long-range dependence to improve the accuracy of predictions of traffic levels in the network to develop congestion control algorithms for the ABR service [10]. Predictors are developed using the stochastic structure of the self-similar background traffic, and the estimated Hurst parameter is important in characterising this structure.

#### 3.1 Modeling ABR Control

The concept of the ABR service is to utilise available bandwidth within the network by controlling the sources which agree to the conditions of the ABR service contract. This means that the network returns control information back to the ABR source regarding the allowable rate which the source can transmit at. The approach which has been used in our research is to determine optimal predictions of future traffic levels, and calculate the ABR rates from these predictions. The network model defined here is based on controlling the bandwidth of the outgoing link by aiming to achieve a specified link utilization. This follows the model proposed by Zhao and Li [14]. The congestion avoidance policy can be formulated as follows,

$$U(k) + R_b(k) = \rho C \quad (5)$$

where the control aim is to keep the offered load at a proportion  $\rho$  ( $0 < \rho < 1$ ) of the outgoing link capacity  $C$ .

The total rate-controlled bandwidth  $U(k)$  is made up of the summation of the individual bandwidths used by the  $N$  rate-controlled connections. Each connection has its own round-trip delay  $\delta_j$  through the network. Then the state variable of the system can be defined as deviation from the target utilization, that is

$$x(k) = \rho C - \left[ \sum_{j=1}^N u_j(k - \delta_j) + R_b(k) \right] \quad (6)$$

We can further define the variable  $W(k) = \rho C - R_b(k)$ , thus resulting in the equation

$$x(k) = - \sum_{j=1}^N u_j(k - \delta_j) + W(k) \quad (7)$$

Equation (7) is in the form of a multi-input single-output (MISO) control system. The aim of this control system is to determine the inputs  $u_j(k)$  so that the variance  $E[x^2(k)]$  is minimized. However, a control system for a MISO system is computationally too expensive. To simplify the system, the available bandwidth is equally shared among rate-controlled connections. Thus, we can define:

$$w_j(k) = \frac{W(k)}{N}, j = 1, \dots, N \quad (8)$$

The system now becomes a collection of  $N$  subsystems, each with their own controller:

$$x_j(k) = -u_j(k - \delta_1) + w_j(k), j = 1, \dots, N \quad (9)$$

where the round-trip delays have been ordered such that  $\delta_1 \geq \delta_2 \geq \dots \geq \delta_N$  without loss of generality.

Our control aim now is to minimize  $E[x_j^2(k)]$ . This is equivalent (refer to [15]) to requiring  $\hat{x}_j(k) = 0$  for all  $k$ . Taking the expectation of (9) and setting  $\hat{x}_j(k) = 0$ , we have the following general control law for each subsystem:

$$u_j(k) = \hat{w}_j(k + \delta_j | W(m) : m \leq k) \quad (10)$$

Thus, we require the allowed rates of the individual source rates  $u_j(k)$  to be equal to the predicted values of the system parameters  $w_j(k + \delta_j)$ , which are determined by the amount of bandwidth available in the outgoing link. This prediction can be achieved by using the self-similarity of the network traffic.

### 3.2 Optimal Prediction of Self-similar Processes

As we have defined our system model in Section 3.1, we require the prediction of the background network traffic which is traversing a particular node in the network to determine the desired rates of the controlled sources. This information experiences a delay  $\delta$  in the network before the effects can be observed at the same node. Hence, we require a  $\delta$ -step predictor. The long-range dependence property of network traffic can be employed to provide more accurate predictions. The optimal linear predictor  $\mathbf{G}_\delta^*$  is of the form:

$$\hat{X}_{k+\delta} = \mathbf{G}_\delta^{*T} \mathbf{X}_M \quad (11)$$

where  $X_k$  is a covariance stationary stochastic process with zero mean, variance  $\sigma^2$  and autocovariance function  $\gamma^*(k)$  and  $\mathbf{X}_M$  is the vector of stored traffic measurements  $\{X_k, X_{k-1}, \dots, X_{k-M+1}\}'$ .  $M$  is the memory-length of the predictor. The solution is given in [16] and is found by taking the expectations  $E(X_{k+\delta} X_{k+\delta-m})$  on both sides of (11) for  $m = k - M + 1, \dots, k$ , resulting in the matrix equation:

$$\mathbf{I}^* \mathbf{G}_\delta^* = \gamma_\delta^* \quad (12)$$



where  $\Gamma^*(i, j) = \gamma^*(i - j)$  is the  $M \times M$  covariance matrix,  $\gamma_\delta^* = (\gamma^*(\delta), \gamma^*(\delta + 1), \dots, \gamma^*(\delta + M - 1))'$  is M-values of the autocovariance function starting at lag  $\delta$  and  $\mathbf{G}_\delta$  is the prediction vector. The variance of the prediction errors is given by

$$v_\delta^* = \gamma^*(0) - \gamma_\delta^{*T} \Gamma^{*-1} \gamma_\delta^* \quad (13)$$

The predicted background traffic levels for traffic with non-zero mean  $\mu_b$  are then calculated as:

$$\hat{R}_b^*(k + \delta) = \mu_b + \mathbf{G}_\delta^{*T} (\mathbf{R}_b(k) - \mu_b) \quad (14)$$

### 3.3 Sensitivity of ABR Control to Hurst Parameter Estimates

Of course, absolute knowledge of the stochastic structure of the background traffic is not possible, and the stochastic structure must be estimated from observations of the traffic. Thus, predictors which are developed for the traffic are inevitably mismatched to some degree to that traffic. This effect of this mismatch can be determined analytical. Consider that the actual predictors are developed from an estimated autocovariance function:

$$\hat{F} \hat{\mathbf{G}}_\delta = \hat{\gamma}_\delta \quad (15)$$

The variance of the prediction errors using this mismatched predictor now becomes

$$\begin{aligned} \hat{v}_\delta &= \mathbf{E} \left[ (R_b(k + \delta) - \hat{R}_b(k + \delta))^2 \right] \\ &= \mathbf{E}[R_b(k + \delta)^2] - 2\mathbf{E}[R_b(k + \delta)\hat{R}_b(k + \delta)] + \mathbf{E}[\hat{R}_b(k + \delta)^2] \\ &= \gamma^*(0) - 2\gamma_\delta^{*T} \hat{F}^{-1} \hat{\gamma}_\delta + \hat{\gamma}_\delta^T \hat{F}^{-1} \Gamma^* \hat{F}^{-1} \hat{\gamma}_\delta \end{aligned} \quad (16)$$

The relative increase in the variance of the prediction errors is

$$K_v = \frac{\hat{v} - v^*}{v^*} \quad (17)$$

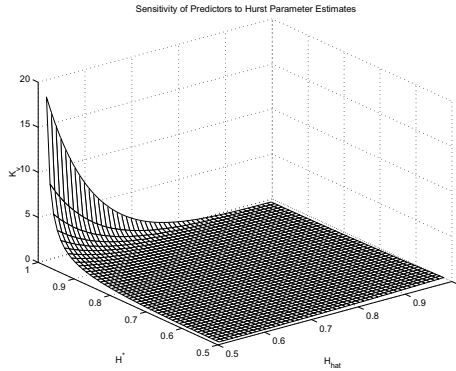
which gives Theorem 1.

**Theorem 1.** *The relative increase in the variance of the prediction errors resulting from a linear predictor which is mismatched to the stochastic structure of the background traffic is given by:*

$$K_v = \frac{\hat{\gamma}_\delta^T \hat{F}^{-1} \Gamma^* \hat{F}^{-1} \hat{\gamma}_\delta + \gamma_\delta^{*T} \Gamma^{*-1} \gamma_\delta^* - 2\gamma_\delta^{*T} \hat{F}^{-1} \hat{\gamma}_\delta}{\gamma^*(0) - \gamma_\delta^{*T} \Gamma^{*-1} \gamma_\delta^*} \quad (18)$$

Using Theorem 1, the sensitivity of congestion control algorithms for the ABR service to the Hurst parameter estimates can be investigated. The fractional Gaussian noise model is used to calculate the relative increase in the variance of the prediction errors when an estimate of the Hurst parameter  $\hat{H}$  is used.

The autocorrelation function for fGn is given in (1), and the relative change  $K_v$  is calculated for pairs of  $(H^*, \hat{H})$ . The resulting surface is shown in Fig. 1. The figure reveals an asymmetry in the sensitivity of the predictors to the Hurst parameter estimates, with an asymptote situated at the point  $(H^*, \hat{H}) = (1.0, 0.5)$ . Thus, the relative increase in variance in the prediction errors rapidly grows when the burstiness of the data becomes more significant, ie. as  $H^* \nearrow 1.0$ , and yet the Hurst parameter is estimated to be close to SRD, ie.  $\hat{H} \searrow 0.5$ . This result emphasizes the importance of recognising the presence of self-similarity within network traffic if it exists. By incorporating the characteristics of self-similarity into resource management algorithms, the impact of the burstiness of self-similar traffic can be avoided.

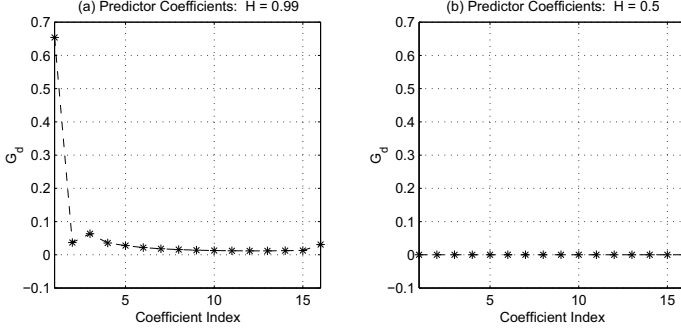


**Fig. 1.** The sensitivity of linear predictors to Hurst parameter estimates using the fractional Gaussian model.

Figure 1 also indicates that another conclusion can be drawn regarding the Hurst parameter sensitivity. The asymmetry at the point diagonally opposite from the asymptote,  $(H^*, \hat{H}) = (0.5, 1.0)$  suggests that it is actually beneficial to err on the side of over-estimating the Hurst parameter and to assume a greater burstiness than may actually exist within the traffic. While this may be intuitively appealing, it does demand an explanation.

Predictors for the boundary values for the Hurst parameter are shown in 2. From the predictors, it is clear why the asymmetry occurs and the reason for its orientation. When  $H^* = 0.99$ , the optimal predictor is shown in Fig. 2(a) where significant weight is given to previous samples because of the LRD effect. However, if the predictor in Fig. 2(b) is used, no weight is given to the previous samples (even though there is a significant amount of information in these samples). This results in significant prediction errors. Consider the other situation, where  $H^* = 0.5$  represents in normally distributed white Gaussian noise, and the mismatched predictor in Fig. 2(a) is used rather than the one in Fig. 2(b). In this case, there is no information about the future sample in the stored data samples and the optimal predictor gives no weight to any of the

samples. When the mismatched predictor in Fig. 2(a) is used, weight is given to these samples, but since the burstiness of the samples is minimal ( $H^* = 0.5$ ), each individual sample does not impact the weighted sum as significantly and overall the prediction errors are comparatively smaller.



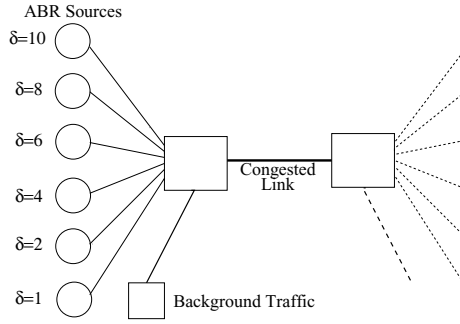
**Fig. 2.** Predictors for the boundary Hurst parameter values: (a)  $\hat{H} = 0.99$ , (b)  $\hat{H} = 0.5$ .

## 4 Simulation Results

The asymmetry of the sensitivity of ABR controllers with respect to the Hurst parameter estimates is investigated in this Section using data sets, giving simulation results which we can compare with the analytical results. The simulation model of the ABR system is shown in Fig. 3, and it consists of six ABR sources competing for the use of a congested link. Each source has its own round-trip delay, and the sources are ordered according to the delay  $\delta_j = \{10, 8, 6, 4, 2, 1\}$ . The congested link is also carrying non-ABR background traffic, which uses a significant proportion of the link capacity, in this case 50% of the capacity on average.

The background traffic is modelled by data sets which consist of ten aggregated VBR data sets [17,18] which have been randomly shifted in time and have been filtered to remove GOP correlation structure of the VBR data. The sources of these aggregated data sets and the Hurst parameter estimates for the data sets are summarised in Table 1. Two additional data sets (the *Combination VBR Data* and *White Gaussian Noise* sets) were used for comparison. The *Combination VBR Data* was produced by aggregating all the previous VBR data sets sources in the table, which had been randomly shifted in time. Finally the *White Gaussian Noise* data set was generated from a normally distributed random number generator, and transformed so that it has an equivalent mean and variance to the *Star Wars* data set.

Comparing the estimates from the different estimation techniques, it is observed that while there is sufficient agreement between the techniques that self-similarity is evident in each of the aggregated VBR data sets and that, for most



**Fig. 3.** The network model for the simulation study.

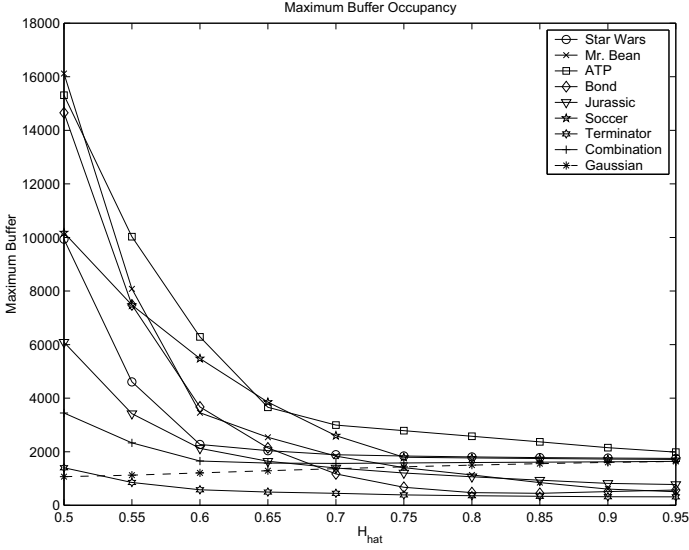
cases, the confidence 95% generated from the AV estimator contains all three estimates, there is also sufficient discrepancy between the values for the Hurst parameter to leave uncertainty in the choice of the value when an ABR congestion control algorithm is being developed. To determine the effect of this uncertainty on the performance of the ABR system, predictors for the simulation model were developed across the entire interval of possible estimates,  $\hat{H} \in [0.5, 1.0)$  and the system was simulated using each predictor.

Buffers at network nodes are designed to handle traffic overflows which occur in the network. In our case, these overflows represent positive prediction errors. The maximum queue length is used as measures of the performance of the network, and an indicator of the sensitivity of the control algorithms to the Hurst parameter estimate under the assumption of infinite network buffers. For

**Table 1.** Comparison between the Hurst parameter estimates for the variance-time (V-T), R/S statistic (R/S) and Abry-Veitch (AV) estimation methods.

Data Set Source	V-T Estimate	R/S Estimate	AV Estimate with 95% CI
Star Wars	0.84	0.90	0.835 [0.774, 0.897]
Mr. Bean	0.76	0.86	0.817 [0.634, 1.000]
James Bond: Goldfinger	0.89	0.86	0.851 [0.669, 1.034]
Jurassic Park	0.79	0.81	0.850 [0.6667, 1.033]
Terminator II	0.77	0.83	0.838 [0.732, 0.943]
ATP Tennis Final 94: Becker - Sampras	0.77	0.85	0.884 [0.778, 0.989]
Soccer World Cup Final 94: Brazil - Italy	0.57	0.79	0.791 [0.685, 0.896]
Combination VBR Data	0.71	0.85	0.811 [0.629, 0.994]
White Gaussian Noise	0.47	0.51	0.495 [0.469, 0.522]

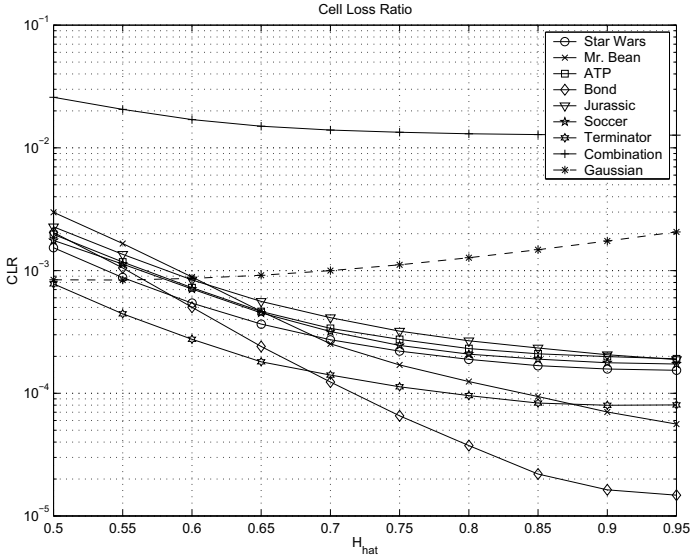
finite buffers, the cell loss ratio (CLR) is used as the performance metric. These simulation results are shown in Figs . 4 and 5.



**Fig. 4.** The maximum queue length at the node against the Hurst parameter estimate.

It is observed from these figures that the performance of the ABR system is highly sensitive to the Hurst parameter estimate with self-similar background traffic. This occurs in the region of the interval  $\hat{H} = [0.5, 1.0)$  where  $\hat{H}$  is close to the 0.5 value, which means that the design has assumed short-range dependence within the network traffic. This sensitivity is revealed in longer queues in the nodes, and greater cell losses. The sensitivity of the system is significantly reduced when the estimates for the Hurst parameter are in the interval  $\hat{H} \in (0.75, 1.0)$ . Comparing these simulation results with the surface for predictor sensitivity derived analytically (Fig. 1), we have further confirmation that the actual values of the Hurst parameters for this type of data is in the interval  $\hat{H} \in (0.75, 1.0)$ .

The *Combination VBR Data* and *White Gaussian Noise* data sets provide an interesting comparison. In both cases the overall sensitivity to  $\hat{H}$  is significantly less than in the other cases. In the case of the *White Gaussian Noise* data set, this reduction in sensitivity agrees with the conclusions from our analytical work, that there is insignificant change in network performance when a mismatched filter is used with SRD traffic. The slight increase in the maximum queue lengths and CLR as the Hurst parameter is increased across the interval  $[0.5, 1.0)$  confirms that the true value for the Hurst parameter is close to 0.5. In the case of the *Combination VBR Data* set, while the sensitivity is not as significant as the other VBR data sets, it still appears that the assumption of self-similarity and



**Fig. 5.** The Cell Loss Ratio, for the system with a finite buffer of maximum capacity 100 cells, versus the Hurst parameter estimate.

a higher estimate for the Hurst parameter does result in marginally improved network performance.

While our simulation results do not prove self-similarity for the aggregated VBR data sets (which is not the purpose of this research), two main conclusions can be drawn from these results. Firstly, significant losses will occur within the ABR control system if the self-similarity of network traffic is not accounted for in the design of the control algorithms. These losses in the system can be reduced by incorporating the characteristics of self-similarity into the algorithms. Secondly, if two estimates  $\hat{H}_1$  and  $\hat{H}_2$  have been obtained for the Hurst parameter of the network traffic, where  $\hat{H}_1 \leq \hat{H}_2$ , then it is more judicious to choose the greater estimate  $\hat{H}_2$  when designing the ABR control algorithm, provided that  $\hat{H}_2 < 1.0$ .

## 5 Conclusions

There is an inevitable uncertainty in the specific value which is assigned to the Hurst parameter when research moves from attempting to prove self-similarity to utilising its characteristics within network architecture. This issue of the impact of the Hurst parameter value has been addressed here both analytically and through simulations. The analytical study revealed an asymmetry in the sensitivity of the predictors to the estimate, especially in the region where the Hurst parameter was significantly underestimated and the background traffic was assumed to be SRD when in reality it was significantly LRD in character. The sensitivity was also addressed using simulations of an ABR control system

using aggregated VBR data sets to model background network traffic. Previous research has statistically proven the self-similarity of the data used, and our own Hurst parameter estimates agreed with that conclusion. The simulations investigated the performance of network buffers when a set of congestion control algorithms were used which were derived from Hurst parameter values that spanned the interval  $\hat{H} \in [0.5, 1.0)$ . The results of our simulations confirmed the analytical study.

This research work has confirmed previous conclusions regarding self-similarity in network traffic, specifically that there is significant impact on the network performance if the self-similar nature of traffic is ignored. Thus, it is important to continue to develop and improve Hurst parameter estimators, and to investigate methods of incorporating the characteristics of self-similarity into network management techniques. In addition, this research demonstrated that, when given a set of estimates of the Hurst parameters, the most judicious choice of the Hurst parameter value is the highest one as this ensures the best network performance.

## Acknowledgements

The authors wish to acknowledge the financial support of Telecom New Zealand (Research Grant E4567).

## References

1. J. Beran, R. Sherman, M. Taqqu, and W. Willinger, "Long-range dependence in variable-bit-rate video traffic," *IEEE Transactions on Communications*, vol. 43, pp. 1566–1579, Feb/Mar/Apr 1995.
2. W. Leland, M. Taqqu, W. Willinger, and D. Wilson, "On the self-similar nature of ethernet traffic (extended version)," *IEEE/ACM Transactions on Networking*, vol. 2, pp. 1–15, Feb. 1994.
3. J. Beran, *Statistics for Long-memory Processes*. Chapman & Hall, New York, 1994.
4. D. Veitch and P. Abry, "A wavelet based joint estimator of the parameters of long-range dependence," *IEEE Transactions on Information Theory*, vol. 45, pp. 878–898, Apr. 1999.
5. I. Norros, "A storage model with self-similar input," *Queueing Systems*, vol. 16, pp. 387–396, 1994.
6. D. Heyman and T. Lakshmann, "What are the implications of long-range dependence for VBR-video traffic engineering?," *IEEE/ACM Transactions on Networking*, vol. 4, pp. 301–317, June 1996.
7. S. Giordano, M. Pagano, R. Pannocchia, and F. Russo, "A new call admission control scheme based on the self similar nature of multimedia traffic," *Proceedings of ICC '96*, Mar. 1996.
8. A. Adas and A. Mukherjee, "On resource management and QoS guarantees for long range dependent traffic," *Proceedings of INFOCOM '95*, pp. 779–787, 1995.
9. G. Chiruvolu and R. Sankar, "An approach towards resource management and transportation of VBR video traffic," *Proceedings of ICC '97*, pp. 550–554, 1997.
10. S. Östring, H. Sirisena, and I. Hudson, "Dual dimensional ABR control scheme using predictive filtering of self-similar traffic," *Proceedings of ICC '99*, June 1999.

11. S. Östring, H. Sirisena, and I. Hudson, "Adaptive ABR congestion control algorithm for non-stationary self-similar traffic," *Submitted to IEEE ATM Workshop '00*, June 2000.
12. G. Gripenberg and I. Norros, "On the prediction of fractional brownian motion," *Journal of Applied Probability*, vol. 33, pp. 400–410, 1996.
13. M. Roughan, D. Veitch, and P. Abry, "On-line estimation of the parameters of long-range dependence," *Proceedings of GLOBECOM '98*, Nov. 1998.
14. Y. Zhao, S. Q. Li, and S. Sigarto, "A linear dynamic model for design of stable explicit-rate ABR control scheme," *Proceedings of INFOCOM '97*, pp. 283–292, Apr. 1997.
15. K. Åström and B. Wittenmark, *Computer-Controlled Systems Theory and Design*. Prentice Hall, New Jersey, 3rd ed., 1997.
16. P. Brockwell and R. Davis, *Time Series: Theory and Methods*. Springer-Verlag, New York, 2nd ed., 1991.
17. O. Rose, "Statistical properties of MPEG video traffic and their impact on traffic modelling in ATM systems," *Proceedings of 20th Conference on Local Computer Networks 1995*, pp. 397–406, Oct. 1995.
18. M. Garrett and W. Willinger, "Analysis, modelling and generation of self-similar VBR video traffic," *Computer Communications Review Proceedings of ACM SIGCOMM*, vol. 24, pp. 269–280, Aug. 1994.



# Providing GFR Guarantees for TCP/IP Traffic over APON Access Systems

Sašo Stojanovski<sup>1</sup>, Maurice Gagnaire<sup>1</sup>, and Rudy Hoebeke<sup>2</sup>

<sup>1</sup> E.N.S.T. InfRés, 46, rue Barrault,  
75634 Paris cedex 13, France  
{sassos, gagnaire}@inf.enst.fr

<sup>2</sup> Alcatel Corporate Research Centre,  
Francis Wellesplein 1, B-2018 Antwerpen, Belgium  
rudy.hoebeke@alcatel.be

**Abstract.** The paper focuses on providing bandwidth guarantees for the Guaranteed Frame Rate (GFR) service category in the context of ATM-based Passive Optical Networks (APONs). Our work builds on the research performed on ATM multiplexers and extends it with the specifics of the APON access system. We study the performance of two known buffer management schemes (DFBA, WFBA), combined with either FIFO or per-VC queueing, when applied on virtual circuits carrying TCP/IP traffic. Our simulation results show that with fairly small buffers at the Optical Network Units, the schemes based on per-VC queueing yield very good performance in large number of cases. The schemes based on FIFO queueing suffer from unfairness, although they do provide bandwidth guarantees in case the GFR bandwidth reservation is kept reasonably small. Tagging helps protecting TCP-like responsive traffic but does not prevent aggressive traffic from grabbing the non-reserved bandwidth. Tagging also increases unfairness.

## 1 Introduction

Guaranteed Frame Rate (GFR) is a new ATM service category intended to support non-real time applications that send data in the form of frames. Contrary to the Available Bit Rate (ABR) capability, the GFR service itself does not provide any explicit feedback to the traffic sources. Therefore, the traffic sources are supposed to be responsive to implicit congestion feedback information (dropped frames) and adapt their transmission rate. Since this is an IP-dominated world, the GFR service category will mostly be used for carrying TCP/IP traffic and the congestion control will consequently be performed via layer-4 mechanisms (TCP), rather than via layer-3 mechanisms (ABR).

The GFR service is defined in [1] with the following parameters: Peak Cell Rate (PCR) and its associated Cell Delay Variation (CDV) tolerance  $\tau_{PCR}$ , Minimum Cell Rate (MCR) and its associated CDV tolerance  $\tau_{MCR}$ , Maximum Burst Size (MBS) and Maximum Frame Size (MFS). The MCR parameter specifies the bandwidth guarantee that should be provided to the user under all

network conditions. However, the user is allowed to send traffic well beyond this guarantee. In the latter case the network may perform some policing actions (tagging or discarding) on the submitted traffic. This is done on a frame-basis rather than cell-basis, which is the main originality of GFR with respect to the existing ATM service categories (notably the nrt-VBR with tagging).

Recently, there has been much interest in mechanisms for supporting the GFR service, notably: buffer management schemes and service disciplines. The former mechanism decides whether a frame is to be admitted into the buffer, whereas the latter determines the order in which the buffered cells will be transmitted.

It is well established (see [2]) that sophisticated per-VC service disciplines (e.g. Weighted Fair Queueing (WFQ) or Weighted Round Robin (WRR)) are sufficient for providing per-connection bandwidth guarantees in situations with infinite buffers, or in situations with traffic compliant to the Generic Cell Rate Algorithm (GCRA). However, if the sources are “elastic” in the sense that they can transmit at rates greater than those contracted (which is the case with GFR) and if the VCs carrying elastic traffic share the same buffer space (which is usually the case in all implementations), then the per-VC service disciplines must be complemented with active buffer management schemes in order to provide bandwidth guarantees. This argument is even stronger when the traffic carried along the ATM VCs is responsive to implicit feedback, as is the traffic generated by TCP sources.

Several GFR-enabling schemes have been investigated in the literature in the context of ATM multiplexers (see [3]), which are normally located at the output ports of ATM switches. Our research extends in the domain of APONs, taking into account their specifics.

An APON access system is shown in Figure 1. It is built on a passive optical tree, with a single Optical Line Termination (OLT) at the root and several Optical Network Units (ONU) at the leaves. It is destined for serving residential and small-business users. The number of ONUs connected to the optical tree is limited by the optical power budget, but the number of end-users may be much larger, since several end-users may be connected to a single ONU.

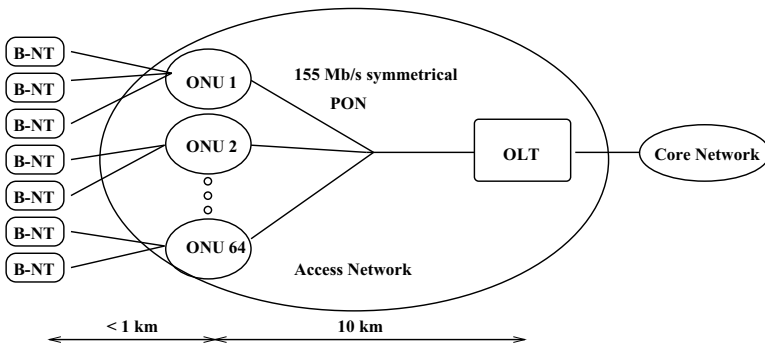


Fig. 1. The APON access system considered in this paper.

Since the APON system is built on a shared medium, a MAC protocol is needed for controlling the access to the medium in the upstream direction. The ATM cells generated by the end-users are buffered at the ONUs. The latter are being periodically polled by the OLT for sending their bandwidth requests. The OLT then issues transmission permits and sends them in the downstream direction. Only upon a reception of a permit is the ONU allowed to transmit a cell. An ideal MAC protocol should be completely transparent to the ATM layer and the whole APON system should appear as an ATM multiplexer. However, this is only partially true for the following reasons: first, the request/permit mechanism introduces an inherent delay (round-trip time of 0.1 ms in our case); second, the system is hierarchical with aggregation occurring at the upper stage, and third, the buffer management schemes are performed at the *input* (the ONUs) rather than the output of the multiplexer.

To our knowledge this is the first study that investigates the performance of TCP/IP traffic carried across APONs. And although the research presented here focuses on a specific kind of ATM access networks, we believe that some of the conclusions hold in general, especially those concerning the use of tagging for protection of TCP-like responsive flows.

In Section 2 we define the buffer management and queueing schemes which we consider for use at the ONU. Then, in Section 3 we propose two scenarios which are investigated in Section 4 by means of simulations. Section 5 summarises our findings.

## 2 Buffer Management and Queueing Schemes

In our study we consider a symmetrical 155.52 Mbit/s APON with 64 ONUs and frame structure as defined in [4]. Although the APON is an example of hierarchical multiplexer, sophisticated hierarchical queueing schemes, such as the one described in [5], cannot be used due to the geographical separation between the two stages. Therefore, two different and independent queueing schemes are performed at each stage. The OLT schedules the bandwidth requests sent by the ONUs using the Weighted Round-Robin (WRR) discipline, defined in [6]. The service is performed on per-ONU rather than per-VC basis, since the OLT has no per-VC visibility. The weights allocated to each ONU are proportional to the aggregated MCR sum of all VCs stemming from the same ONU. On the other side, the ONU has the choice of using either the same WRR discipline, but this time on a per-VC basis, or do FIFO queueing.

We consider two known buffer management schemes at the ONU: Weighted Fair Buffer Allocation (WFBA) and Differential Fair Buffer Allocation (DFBA), defined in [7] and [8], respectively. Both of them make use of two global thresholds: *LBO* and *HBO*, standing for Low and High Buffer Occupancy. A buffer allocation weight is associated to each VC, and typically this weight is proportional to the connection's MCR. Untagged frames (CLP=0) are always accepted if the global buffer occupancy  $X$  is lower than *LBO*. The differences between the two algorithms appear when the global buffer occupancy is between the two

thresholds ( $LBO < X < HBO$ ). Both schemes use an admission criterion which is a function of the global and individual buffer occupancies, as well as the allocated weights. The frames which do not pass the admission criterion in WFBA are dropped in a deterministic manner, whereas in DFBA they are dropped with a probability which is also a function of the above-mentioned criteria. Note that probabilistic dropping is also used in Random Early Detection (RED), the most widely deployed buffer management scheme in today's Internet (see [9]).

In its original version WFBA does not distinguish between tagged and untagged frames. However, this extension is straightforward: tagged frames in both schemes are admitted into the buffer only if the global buffer occupancy  $X$  is lower than  $LBO$ .

Two conformance definitions have been defined for the GFR service: GFR.1 and GFR.2. The difference between the two is that GFR.2 allows the network to tag the excess traffic using the  $F\text{-}GCRA(T, f)$  algorithm, where  $T = \frac{1}{MCR}$  and  $f \geq (MBS - 1) \cdot (\frac{1}{MCR} - \frac{1}{PCR})$ . Note, however, that tagging can also be *virtual* i.e. frames can be tagged internally at the ONU, without actually modifying the CLP bit (see [10]). The virtual tagging allows us to take a unified approach for both GFR.1 and GFR.2 conformance definition.

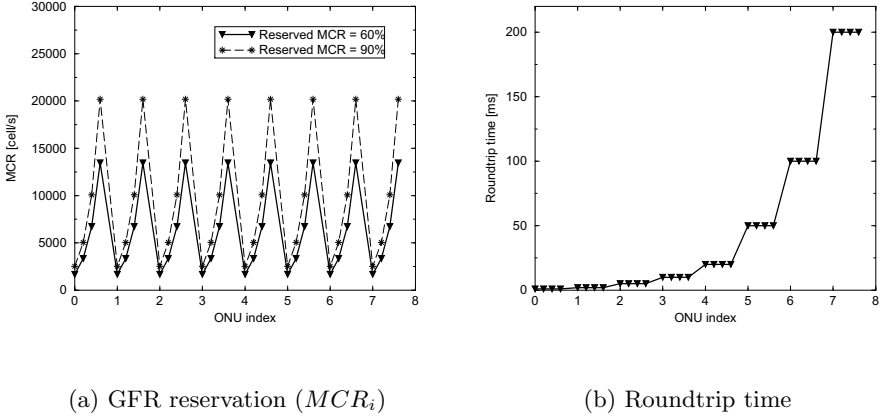
By combining the two buffer management schemes with FIFO or per-VC queueing, we obtain the following four schemes: WFBA+FIFO, DFBA+FIFO, WFBA+WRR and DFBA+WRR. We investigate their performances by means of simulations in the following section.

### 3 Simulation Scenarios

We consider two heterogeneous scenarios referred to as: RESP1 and RESP2. In both scenarios we consider 32 GFR VCs carrying 10 TCP-NewReno connections each. The total number of TCP connections equals 320. Only 8 ONUs (out of 64) are active and each one is traversed by four GFR VCs. The VCs do not have equal MCR reservations, neither RoundTrip Times (RTT). The two scenarios differ in the way the MCRs and RTTs are distributed across the ONUs. We tried to cover as many cases as possible. The parameters which are common and specific to the two scenarios are given in Table 1 and Table 2, respectively. Table 1 contains the TCP-specific parameters, such as: version, Maximum Segment Size or timer granularity. It also shows the ONU buffer threshold settings ( $LBO$  and  $HBO$ ) which are used for buffer management. Table 2 explains the MCR and RTT distribution across the 32 VCs and eight active ONUs.

The scenario-specific parameters (i.e. MCR reservations and RTT) for scenarios RESP1 and RESP 2 are illustrated in Figure 2 and Figure 3, respectively. Figures 4(a) and 4(b) illustrate the bandwidth-delay product for scenario RESP1 and RESP2, respectively, expressed in cells. In these, as well as in all subsequent figures, the abscissa values identify the active ONUs.

We use the NewReno version of TCP which is described in [11]. This is a bug-fixed version of TCP Reno which improves the performance of the latter in

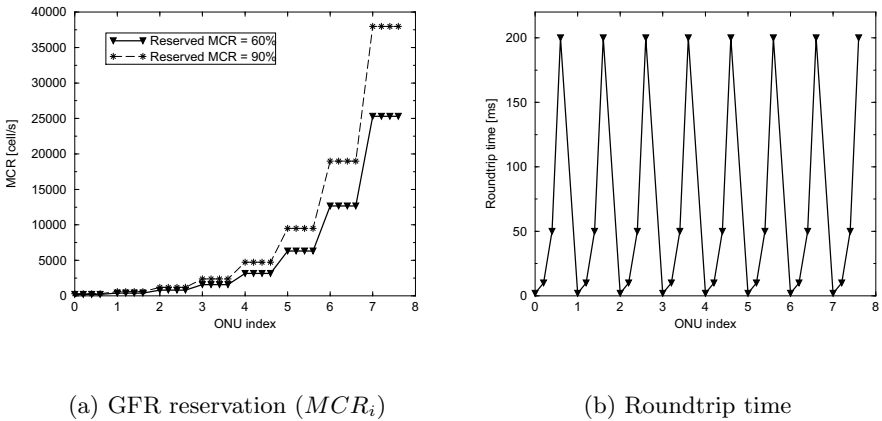


**Fig. 2.** Scenario RESP1.

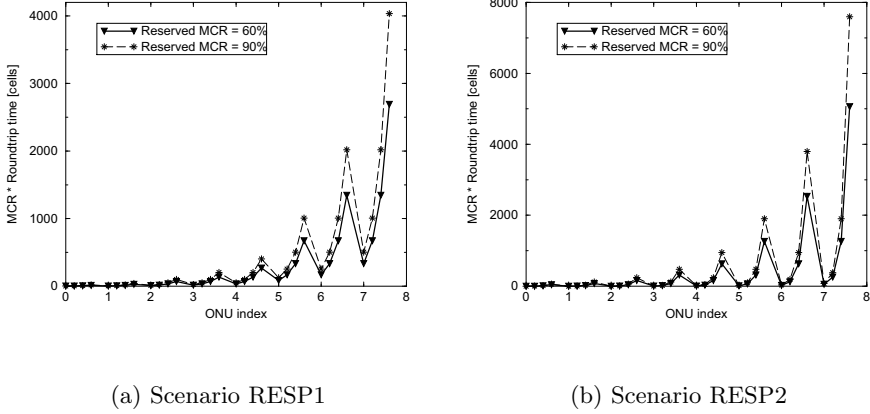
case of multiple segments dropped from a window of data. All 320 TCP sources are persistent i.e. they have always data for transmission.

## 4 Simulation Results

The simulations reported in this paper correspond to 30 seconds of simulated time. The results are expressed via the normalised goodput received by each VC, defined as:  $R = \frac{Goodput_i}{MCR_i}$ .  $R = 1.0$  means that  $VC_i$  has realised goodput which is exactly equal to its reservation  $MCR_i$ , without receiving any excess bandwidth. Similarly,  $R = 2.0$  means that  $VC_i$  has realised goodput which is equal to twice its reservation and  $R = 0$  means that  $VC_i$  has not realised any goodput at all.



**Fig. 3.** Scenario RESP2.

**Fig. 4.** Bandwidth-delay product.**Table 1.** Common parameters for scenarios RESP1 and RESP2

<i>Active ONUs</i>	8 (out of 64)
<i>GFR connections per ONU</i>	4
<i>TCP connections per VC</i>	10
<i>Total MCR reservation</i>	either 60% or 90% of the system's capacity
<i>TCP Max Segment Size (MSS)</i>	1460 octets
<i>Maximum Frame Size (MFS)</i>	32 cells (This corresponds to the chosen MSS with all the overhead: 20 octets (TCP) + 20 (IP) + 8 (LLC/SNAP) + 8 (CPCS-AAL5) + padding.)
<i>ONU buffer size</i>	2000 cells
<i>LBO (HBO) threshold</i>	900 (1800) cells
<i>TCP version</i>	NewReno
<i>TCP timer granularity</i>	10 ms

In subsection 4.1 we consider scenarios with responsive traffic only. No tagging is used. Then, in subsection 4.2 we evaluate the TCP performance in presence of non-responsive and very aggressive traffic. The conclusion of 4.2 is that tagging has to be used. Therefore, in subsection 4.3 we revisit the responsive traffic scenario from 4.1, but this time with tagging being applied.

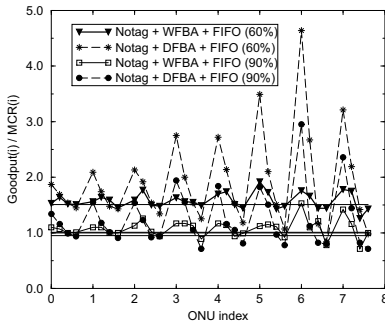
#### 4.1 Responsive Traffic Only and No Tagging

Figure 5 illustrates the normalised goodput for schemes relying on either FIFO or WRR queueing, for scenario RESP1. The global GFR reservation (i.e. the sum of per-connection  $MCR_i$ ) equals either 60% or 90% of the system capacity. Also shown in the figures are three horizontal lines. The first one,  $y = 1.0$ , represents the lowest value for the normalised goodput at which the bandwidth guarantee

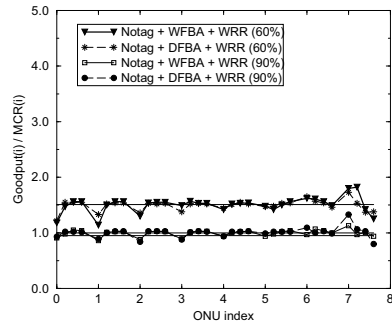
**Table 2.** Specific parameters for scenarios RESP1 and RESP2

	<i>RESP1</i>	<i>RESP2</i>
<i>MCR for VCs at the same ONU</i>	relate to each other as 1 : 2 : 4 : 8	relate to each other as 1 : 1 : 1 : 1
<i>Aggregated MCR of the eight active ONUs</i>	relate to each other as 1 : 1 : 1 : 1 : 1 : 1 : 1 : 1	relate to each other as 1 : 2 : 4 : 8 : 16 : 32 : 64 : 128
<i>Roundtrip time</i>	same for VCs at the same ONU; changes from one ONU to another as: 1 : 2 : 5 : 10 : 20 : 50 : 100 : 200 ms	2 : 10 : 50 : 200 ms for every four consecutive VCs; the same pattern is repeated at each ONU

is still met. The other two lines ( $y = 1.66$  and  $y = 1.11$ ) correspond to the ideal value for the normalised goodput, for which every VC gets a share of the available (non-reserved) bandwidth in proportion to its MCR. (Note:  $1.66 = \frac{100}{60}$  and  $1.11 = \frac{100}{90}$ .)



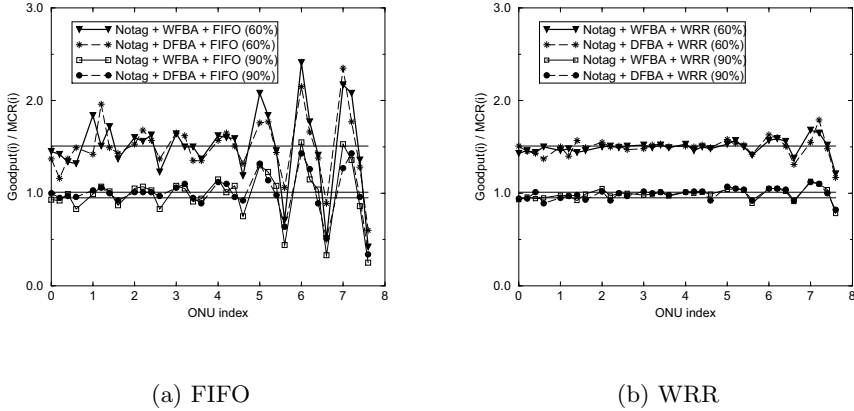
(a) FIFO



(b) WRR

**Fig. 5.** RESP1: normalised TCP goodput.

As seen from Figure 5, when the global GFR reservation equals 60%, all VCs attain their guarantees ( $R > 1.0$ ). When FIFO queueing is used, the available bandwidth is not fairly distributed since low-rate VCs at each ONU get a higher proportion of available bandwidth (the curves contain ripples). This is especially striking with the DFBA scheme. When the global reservation equals 90%, several VCs fail to achieve their guarantee, especially with the DFBA scheme. On the other hand, both buffer management schemes perform equally well when WRR queueing is used. Almost all VCs attain their guarantee and free bandwidth is distributed ideally (almost flat curves).



**Fig. 6.** RESP2: normalised TCP goodput.

Figure 6 illustrates the performance of FIFO and WRR based schemes in scenario RESP2. Again one sees that WRR schemes are fairer than FIFO schemes in distribution of free bandwidth. What is new is that there is no striking difference between WFBA and DFBA when FIFO queueing is used. This is probably due to the fact that VCs belonging to the same ONU in scenario RESP2 have equal MCR reservations.

It can further be noted from Figure 6 that even at 60% reservation there are several connections that do not meet the guarantee when FIFO queueing is used. This is explained by the fact that the last two ONUs contain VCs whose bandwidth-delay product is greater than the ONU buffer size (much greater than in scenario RESP1). On the other hand, the WRR schemes (see again Figure 6) have no problems in meeting the guarantee at 60% reservation. Furthermore, they succeed to make it within 15% of the MCR guarantee at 90% reservation, even for the most extreme case (the fourth VC from  $ONU_7$ ).

Curiously enough, the probabilistic dropping in DFBA does not yield better performance than the deterministic dropping of WFBA. This may be counter-intuitive, since the field trials with RED have shown the benefits of probabilistic dropping when carrying TCP traffic in today's Internet. We see two possible explanations for this observation. First, the ATM VCs in our scenarios carry several TCP connections, so even if frames in WFBA are dropped deterministically, they do not necessarily belong to the same TCP connection. Second, and more importantly, the RED scheme reacts on the *average* rather than *instantaneous* queue size as in DFBA. Hence, even if DFBA drops frames probabilistically, there is still a non-negligible probability of “hitting” the same TCP connection when the global occupancy temporarily persists at higher values. Given this remark, we shall consider only the WFBA scheme in the remainder of this paper.



## 4.2 TCP Performance in Presence of Non-Responsive Traffic

It is easy to show that without tagging, the buffer management schemes themselves are not sufficient for protecting the responsive traffic from the aggressive one. Tagging is an effective way for preventing the aggressive sources from occupying large portion of the shared buffer space. The network elements operating at the ATM layer (e.g. ONUs) have no higher layer visibility and, hence, do not discriminate against responsive and non-responsive traffic (e.g. TCP from UDP). Even if that were the case, the ON U must still not rely on the higher-layer information, since there may be faulty or deliberately aggressive TCP implementations out there. Therefore, tagging should systematically be applied on *all* traffic.

In this subsection we consider a scenario with non-responsive greedy sources and we refer to it as NONRESP2. Scenario NONRESP2 is obtained from RESP2 by substituting the first VC at every active ONU with a VC carrying non-responsive traffic. The latter is modelled as a constant bitrate flow carrying 32-cell frames at 155.52 Mbit/s (measured at the physical layer). Thus the entire system is at extreme overload.

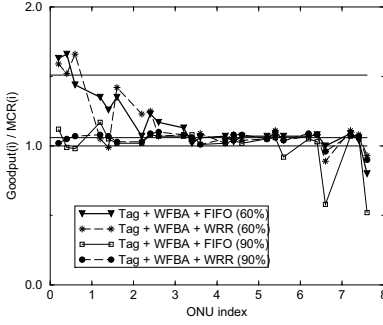
When tagging is applied, there is a choice to be made for the MBS parameter in the F-GCRA function. Several approaches for the MBS exist in the literature. For instance, [12] proposed a default value of 192 cells, whereas [10] proposed to compute it as a function of the MCR and the number of TCP connections in a VC. We propose to use a fixed and rather large value of:  $MBS = 200 \cdot MFS$ , regardless of the connection's MCR.

Figure 7 shows the TCP throughput of the responsive connections in case WFBA+FIFO (or WFBA+WRR) with tagging is used, at both 60% and 90% reservation. The figure shows that with the use of tagging the responsive sources can be protected from the non-responsive ones as far as the MCR guarantee is concerned. The only connections that fail to meet their guarantee are those with extremely large bandwidth-delay product, and this should not be attributed to the presence of aggressive traffic.

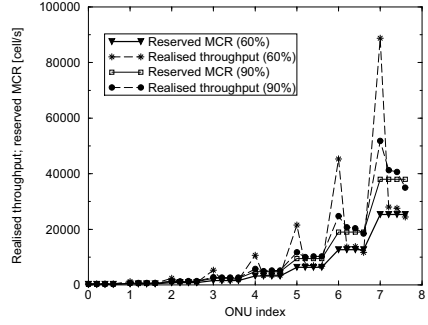
However, the Figure 7 also shows that the normalised goodput curves are collapsed around  $y=1.0$ . This means that, while the bandwidth guarantee is met in the majority of cases, the *free* bandwidth has entirely been grabbed by the aggressive connections. This is clearly visible in Figure 8 which shows the MCR and the realised throughput for each VC (including the aggressive ones). One sees that the greedy connections have realised much better throughput than the responsive ones.

## 4.3 Responsive Traffic with Tagging

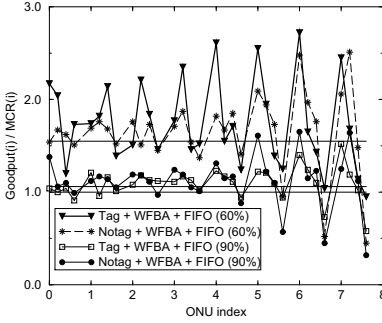
Subsection 4.2 highlighted the importance of tagging. Since the ONUs have no higher layer visibility, they should perform tagging on all traffic. Next we revisit the case with TCP traffic only, which was considered in subsection 4.1, but this time we investigate the impact of tagging, even in absence of non-responsive traffic. Because of lack of space, only RESP2 scenario and WFBA scheme are considered.



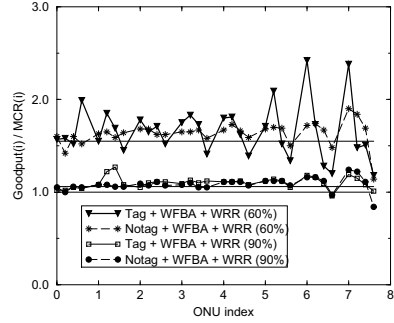
**Fig. 7.** NONRESP2: normalised TCP goodput of responsive VCs under the WFBA+FIFO and WFBA+WRR schemes.



**Fig. 8.** NONRESP2: realised goodput of both responsive and aggressive VCs under the WFBA+WRR scheme.



(a) Tagging + FIFO



(b) Tagging + WRR

**Fig. 9.** RESP2: normalised TCP goodput with tagging.

Figure 9 shows the VC goodput when tagging is used in addition to either WFBA+FIFO or WFBA+WRR. The curves corresponding to the non-tagged case (Figure 6) are also reproduced for comparison. The figures show that at high GFR reservation there is no significant difference whether tagging is used or not. However, at 60% reservation one sees that the distribution of free bandwidth with tagging becomes less fair (indicated by greater ripples in the normalised throughput curves). This is especially apparent when WRR queueing is used.

## 5 Conclusions

In this paper we have considered two buffer management schemes (WFBA and DFBA) combined with two queueing schemes (FIFO and WRR) in order to

provide GFR guarantees to TCP/IP traffic over APON access networks. We find out that:

- at reasonably small reservations (60% of the system capacity), all schemes allow the VCs to attain their guarantees, with the exception of some VCs with extremely high bandwidth-delay product in the FIFO schemes;
- at very high GFR reservation (90% of the system capacity), the high-rate VCs at several ONUs fail to meet their guarantee in the FIFO schemes, whereas in the WRR schemes this happens only for VCs with extremely high bandwidth-delay product;
- the FIFO-based schemes are intrinsically unfair in free bandwidth distribution, since low-rate VCs get larger proportion of excess bandwidth;
- the schemes with per-VC queueing distribute the free bandwidth in a very fair manner, except for VCs with extremely high bandwidth-delay product;
- the probabilistic dropping of DFBA does not bring any performance gain compared to the less complicated WFBA scheme;
- tagging is necessary in order to isolate responsive traffic from aggressive one;
- tagging increases unfairness in absence of non-responsive traffic, and
- tagging does not prevent aggressive traffic from grabbing all the free bandwidth.

Referring to the last conclusion, it would be interesting to investigate whether the three-colour tagging, which is currently considered in the diffserv community (see [13]) would help. Although the ATM cell may have only two colours (CLP=0 or CLP=1), three colours could still be implemented via virtual tagging.

## Acknowledgements

This work has been sponsored by Alcatel CRC in Antwerp-Belgium within a research contract with ENST.

## References

1. ATM Forum's Traffic Management Specification 4.1, *ATM Forum af-tm-0121.000* (1999)
2. B.Suter, T.V.Lakshman, D.Stiliadis, A.Choudhury, "Efficient Active Queue Management for Internet Routers", *Network+Interop98 Engineers Conference Notes, Las Vegas* (1998)
3. Olivier Bonaventure, "Providing bandwidth guarantees to internetwork traffic in ATM networks", *Proceedings on ATM '98 Workshop, Fairfax* (1998)
4. ITU-T Recommendation G.983, "High Speed Optical Access Systems Based on Passive Optical Network (PON) Techniques" (1998)
5. J.C.R. Bennet and H.Zhang, "Hierarchical Packet Fair Queueing Algorithms", *IEEE/ACM Transactions on Networking, vol.5, no.5* (1998)
6. Manolis Katevenis, Stefanos Sidiropoulos and Costas Courcoubetis, "Weighted Round Robin Cell Multiplexing", *IEEE JSAC, vol.9, no.8* (1991)

7. Juha Heinanen and Kalevi Kilkki, "A Fair Buffer Allocation Scheme", *Computer Communications*, vol.21, no.3 (1998)
8. Rohit Goyal, "Buffer Management for the GFR Service", *ATM Forum contribution af98-0405* (1998)
9. Braden, Clark et al., "Recommendation on Queue Management and Congestion Avoidance in the Internet", IETF RFC 2309 (1998)
10. Byoung-Joon Lee, "GFR Dimensioning for TCP Traffic", *ATM Forum contribution af98-0271* (1998)
11. Tom Henderson and Sally Floyd, "The NewReno Modification to TCP's Fast Recovery Algorithm", *Internet Draft draft-ietf-tcpimpl-newreno* (1998)
12. Juha Heinanen, "GFR Signalling", *ATM Forum contribution af98-0024* (1998)
13. Juha Heinanen, Roch Guerin, "A Three-Colour Marker", *Internet Draft draft-heinanen-diffserv-tcm-01.txt* (1999)

# Buffer Size Requirements for Delay Sensitive Traffic Considering Discrete Effects and Service-Latency in ATM Switches

Steven Wright<sup>1</sup> and Yannis Viniotis<sup>2</sup>

<sup>1</sup> BellSouth Science & Technology, 41G70 BSC, 675 West Peachtree St NE,  
Atlanta, GA 30375 USA

Steven.wright@snt.bellsouth.com

<sup>2</sup> Dept. ECE, North Carolina State University, Raleigh , NC 27695-7911,USA.  
candice@eos.ncsu.edu

**Abstract.** Various approaches to buffer size and management for output buffering in ATM switches supporting delay sensitive traffic are reviewed. Discrete worst case arrival and service functions are presented. Using this format, bounds are developed for buffer size under zero cell loss for leaky bucket constrained sources. Tight bounds are developed for the case of discrete arrival functions with fluid servers and fluid arrival functions with discrete servers. A bound on the buffer size is also proposed for the case of discrete arrival and service process. While this bound is not exact, the maximum gain that could be achieved by a tighter bound is bounded. In some cases it is possible to reduce the buffer size requirements through over allocation of link bandwidth. Feasibility conditions for this scenario are developed.

## 1 Introduction

ATM network support [13] for delay-sensitive QoS requirements requires ATM switches to support the QoS guarantees. Low service-latency schedulers, e.g. RRR [4] provide a mechanism for supporting delay QoS guarantees if sufficient switch resources - including buffer space and bandwidth - are available. The focus of this paper is the buffer requirements rather than the scheduler design. The buffer size constraints switch performance (e.g. Connection Admission Control) while representing a significant fraction of interface costs for wire speed interfaces. This paper develops expressions for the buffer size requirements (see equations (1),(11),(12)and (13)) that can be used in buffer optimization problems for specific schedulers, particularly simple Latency-Rate ( $\mathcal{L}\mathcal{R}$ ) schedulers (see [10]) derived from Weighted Round Robin. These expressions are used to explore potential buffer reductions. In contrast to other buffer studies (see [6],[8],[9]), we provide more precise formulations for the buffer requirements due to discrete effects in the arrival and service processes as well as considering service-latency. The paper also provides quantified examples with a specific WRR scheduler to achieve reductions in buffer size requirements. We assume a simple output buffering arrangement such as that used for Burstiness-Class based Queuing (B-CBQ, see [14]), where several queues ( $\alpha, \beta, \chi$ ) are serviced by an  $\mathcal{L}\mathcal{R}$  scheduler which provides each queue with a guaranteed

bandwidth and (potentially) delay bound guarantees. The buffer size formulations in this paper focus on the buffer requirements for an individual queue. Multiple Pt-Pt or MPt-Pt ATM connections are allocated to a queue (i.e. connections  $VC_{\alpha,1..n}$  are directed to queue  $\alpha$ ).  $\mathcal{LR}$  Delay bound guarantees require source traffic that is leaky bucket  $(\sigma, \rho)$  constrained with a worst case burst of size  $\sigma$ , and arrival rate  $\rho$ . In this paper, we assume worst-case aggregate leaky-bucket-constrained arrival function at a queue, i.e., periodic bursts of size  $\sigma$  (see e.g. [3]). The buffer is cleared with a period  $T$  given by the ratio  $\sigma/\rho$ . We are particularly interested in the case where the buffer requirements can be reduced from the maximum burst size ( $\sigma$ ). Buffer size requirements are typically considered at switch design time, however for some switch designs buffers may be reallocated between queues while the switch is operational. For such situations, the computational complexity of estimating buffer requirements is an important issue. The paper is organized as follows. Section 2 places this work in the context of prior work. The buffer size, assuming fluid models for arrivals and service, and also considering service-latency, is considered in section 3. The buffer size requirements under discrete arrival and/or service functions (again considering service-latency) are discussed in section 4. Section 5 explores numerically the feasibility and magnitude of buffer size reductions possible by rate over-allocation for WRR schedulers. Conclusions are provided in section 6.

## 2 Prior Work

Previous work had focussed largely on ideal fluid arrival and service processes (e.g. WFQ) or had simply assumed that the allocated rate matched in the requested rate exactly, rather than considering some potential for over allocation of bandwidth in order to minimize buffer occupancy. The absolute delay bounds both for the GPS schedulers and the more generic version for  $\mathcal{LR}$  schedulers [10] rely on a maximum buffer size and a guaranteed service rate in order to produce the delay bound. The tradeoff of additional guaranteed rate for reduced buffer requirements was identified by [3] in the context of work on equivalent bandwidth capacity. They developed an approach to buffer allocation for ATM networks which reduced the two resource allocation problems (buffer & bandwidth) to a single resource allocation (“effective bandwidth”) problem. [8] built on the work of [3] and separated the buffer/bandwidth tradeoff into two independent resource allocation problems. While work on scheduler design (e.g. [4], [10]) has identified several issues related to service-latency, the impact on the buffer-bandwidth tradeoff has not been explicitly considered. The theory related to fluid models of arrival and service curves has recently been extended into an elegant network calculus (e.g., [1], [2], [7]). While these approaches typically are used to derive end-to-end network delay bounds, they can also be used to provide assertions regarding buffering requirements. In the realm of equipment design, the service curves, and network calculus, are an intermediate form and equipment optimizations must be recast in terms of scheduler design parameters. In practice, ATM switches must deal with discrete data units (cells) and consider the effects of service-latency on buffer requirements. Previous work on discrete buffer sizing for

WRR has been largely empirical (e.g. [9], [6], [5]) and addressed towards loss based QoS parameters, rather than delay-sensitive QoS requirements.

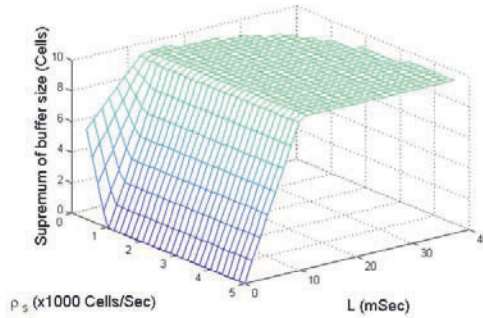
### 3 Buffer Size with Fluid Arrival and Fluid Service with Service-Latency

The main result for the buffer size ( $b$ ) in the case of fluid arrivals and service processes is given in equation (1). Three cases must be considered:

- if the service-latency ( $L$ ) is such that service starts after arrivals have reached the maximum value (which occurs at  $t=\tau_\sigma$ ), then the supremum occurs at  $t=\tau_\sigma$  and  $b=\sigma$ , and no reduction in buffer requirements is possible.
- if the arrival rate ( $\rho_a$ ) is less than (or equal to) the service rate ( $\rho_s$ ), then the supremum occurs at  $t=L$ , and  $b=\rho_a L$ .
- if the arrival rate is greater than the service rate, then the supremum occurs at  $t=\tau_\sigma$  and  $b=\sigma-\rho_s(\tau_\sigma-L)$ .

$$b \geq \begin{cases} \sigma & \text{if } \{L \geq \tau_\sigma\} \\ \sigma - \rho_s(\tau_\sigma - L) & \text{if } \{L < \tau_\sigma\} \text{ and if } \{\rho_a > \rho_s\} \\ \rho_a L & \text{if } \{L < \tau_\sigma\} \text{ and if } \{\rho_a \leq \rho_s\} \end{cases} \quad (1)$$

An example of the buffer space requirements is illustrated in Fig. 1 using equation (1). In order to reduce the buffer requirements below  $\sigma$ , the service-latency must be less than  $\sigma/\rho_a$ , (10msec in this example). Simply increasing the service rate may not reduce the buffer requirements below  $\sigma$ . Even for the cases where  $L < 10\text{msec}$ , increasing the service rate beyond  $\rho_a$  provides no additional benefit which is consistent with the results of [8]. If the service rate,  $\rho_s$ , is less than the peak arrival rate,  $\rho_a$ , then only smaller reductions from  $\sigma$  are possible. While the potential reduction of 10 cells for the connection in this example may not seem significant, we recall that there may be several thousand connections on an ATM interface at OC-3 or higher rates. Also, the burst size, used in this example is very small for VBR traffic.



**Fig. 1** Buffer Space Requirements for Fluid Arrivals and Fluid Service with Service-Latency. The parameters for the example are:  $\sigma = 10$  Cells;  $\rho = 250$  Cells/Sec;  $\rho_a = 1000$  Cells/Sec;  $500 \leq \rho_s \leq 5000$  Cells/Sec;  $0 \leq L \leq 40$  msec.

## 4 Effect of Discrete Arrivals and Discrete Service on Buffer Size

The fluid model ignores discrete effects in the arrival and service functions. Significant buffer reductions are still possible, even after allowing for discrete effects in the arrival and service functions. In contrast to the fluid arrival functions typically based on two  $(\sigma, \rho)$  or three  $(\sigma, \rho, \rho_a)$  leaky bucket parameters, we use a worst-case leaky-bucket discrete arrival function based on four parameters— $(\sigma, \rho, \rho_a, k)$ ; where  $k$  is the step size in the same data units as  $\sigma$  (see Fig. 4). Similarly, we need to move from the two-parameter  $(L, \rho_s)$  fluid service function of section 0 to a worst-case discrete service function based on three parameters  $(L, \rho_s, m)$  where  $m$  is the service step size in the same data units as  $\sigma$  (see Fig. 2). The main result of this section is the formulation for the discrete arrival function in equation (4) and for the discrete service function in equation (8). Also presented are a pair of fluid functions that bound each of these discrete functions. These are equations (5) and (6) for the bounds on the discrete arrival function and equations (9) and (10) for the bounds on the discrete service function.

### 4.1 Arrival Functions

The incoming arrivals are not eligible for service until the time,  $\tau_a$ , given by equation (2). Equation (3) illustrates the time,  $\tau_\sigma$ , at which the maximum arrival burst,  $\sigma$ , is reached. The worst-case discrete arrival function is then defined by equation (4). We can consider the discrete arrival function as bounded by two fluid functions:  $A_{\max}(t)$  (refer Equation (5)) and  $A_{\min}(t)$  (refer equation (6)). The maximum error in the bounding functions is one step,  $k$ . As  $k \rightarrow 0$ , the discrete model is less relevant and  $A(t)$  converges towards the  $A_{\min}(t)$  function.

$$\tau_a = \frac{k}{\rho_a} \quad (2)$$

$$\tau_\sigma = \tau_a \left\lceil \frac{\sigma - k}{k} \right\rceil \text{ for } \sigma \geq k. \quad (3)$$

$$A(t) = \begin{cases} 0 & t \leq 0 \\ k \left( 1 + \left\lfloor \frac{t}{\tau_a} \right\rfloor \right) & 0 \leq t < \tau_\sigma \\ \sigma & \tau_\sigma \leq t \leq T \end{cases} \quad (4)$$

$$A_{\max}(t) = \begin{cases} 0 & t \leq 0 \\ k + \rho_a t & 0 \leq t < \tau_\sigma \\ \sigma & \tau_\sigma \leq t \leq T \end{cases} \quad (5)$$

$$A_{\min}(t) = \begin{cases} 0 & t \leq 0 \\ \rho_a t & 0 \leq t < \tau_\sigma \\ \sigma & \tau_\sigma \leq t \leq T \end{cases} \quad (6)$$



## 4.2 Service Functions

The generalized worst case discrete service function is shown in Fig. 2, where service proceeds in discrete steps up until the maximum buffer occupancy has been served. Many link schedulers offer rate guarantees over a longer time-scale than one cell transmission time, and service some quanta ( $m > 1$  cell) of data at a step [11]. The period,  $\tau_s$ , associated with a service step is given by equation (7). We chose to separate the effects and leave  $L$  to reflect service-latency associated with the start of the service function beyond the scheduling rate granularity. This formulation reflects that buffers are freed at the end of the service time-scale. A discrete version of the service function is then given by equation (8). The worst case discrete service function is bounded by two fluid service functions:  $S_{\max}(t)$  (refer equation (9)) and  $S_{\min}(t)$  (refer equation (10)). The maximum difference between in the bounds is  $m$ . As  $m \rightarrow 0, \tau_s \rightarrow 0$  and the discrete model becomes less relevant as it converges towards  $S_{\max}(t)$ .

$$\tau_s = \frac{m}{\rho_s} \quad (7)$$

$$S(t) = \begin{cases} 0 & 0 \leq t \leq L \\ \min(A(t-L), m \left\lfloor \frac{t-L}{\tau_s} \right\rfloor) & L \leq t \leq T \end{cases} \quad (8)$$

$$S_{\max}(t) = \begin{cases} 0 & 0 \leq t \leq L \\ \min(A(t-L), \rho_s(t-L)) & L \leq t \leq T \end{cases} \quad (9)$$

$$S_{\min}(t) = \begin{cases} 0 & 0 \leq t \leq L + \tau_s \\ \min(A(t-L-\tau_s), \rho_s(t-L-\tau_s)) & L + \tau_s \leq t \leq T \end{cases} \quad (10)$$

## 4.3 Fluid Arrivals and Discrete Service

Consider the fluid arrival process and discrete service process with service-latency in Fig. 2(a). Using the discrete service function may provide an improved result (a potential reduction up to  $m$ ) if we can evaluate the supremum. The main result is presented in equation (11) (refer to [12] for the proof). In brief, the intuition on locating the supremum from the fluid case is extended by whether the supremum occurs at the first step in the service function after  $t=L$ , i.e.  $t=\tau_2$ , or at the last step in the service function prior to  $t=\tau_\sigma$  i.e.  $t=\tau_5$ . An example of the per-connection buffer requirements for the case of  $m=3$  is shown in Fig. 2(b), where the supremum occurs at  $\tau_2$  (~6msec). The arrival function used was a fluid arrival with parameters  $\{\sigma = 10 \text{ Cells}; \rho = 250 \text{ Cells/Sec}; \rho_a = 1,000 \text{ Cells/Sec}\}$ .

$$b \geq \begin{cases} \sigma & \text{if } \{\tau_2 \geq \tau_\sigma\} \\ \sigma - m \left\lfloor \frac{\tau_\sigma - L}{\tau_s} \right\rfloor & \text{if } \{\tau_2 < \tau_\sigma\} \text{ and if } \{\rho_a > \rho_s\} \text{ and if } m \leq \sigma - \rho_a \tau_s \\ \rho_a \tau_s - m \left\lfloor \frac{\tau_\sigma - L}{\tau_s} \right\rfloor + m & \text{if } \{\tau_2 < \tau_\sigma\} \text{ and if } \{\rho_a > \rho_s\} \text{ and if } m > \sigma - \rho_a \tau_s \\ \tau_2 \rho_a & \text{if } \{\tau_2 < \tau_\sigma\} \text{ and if } \{\rho_a \leq \rho_s\} \end{cases} \quad (11)$$

At low latencies, when the service rate exceeds the peak arrival rate, the buffer requirements can be significantly reduced. With  $\rho_s = \rho_a = 1000$  cells/sec in Fig. 3, the buffer requirements are reduced to 3-8 cells depending on the latency of 0-5mS. This is a reduction of 20-70% from a buffer size based on the peak burst size,  $\sigma$  (=10 cells in this example). When the service rate is below the peak arrival rate, the effects of increasing service-latency are impacted by the service step ( $m=3$ ) as illustrated by the series of plateaus in Fig. 3. These are not seen in the fluid arrival and service model shown in the example of Fig. 1.

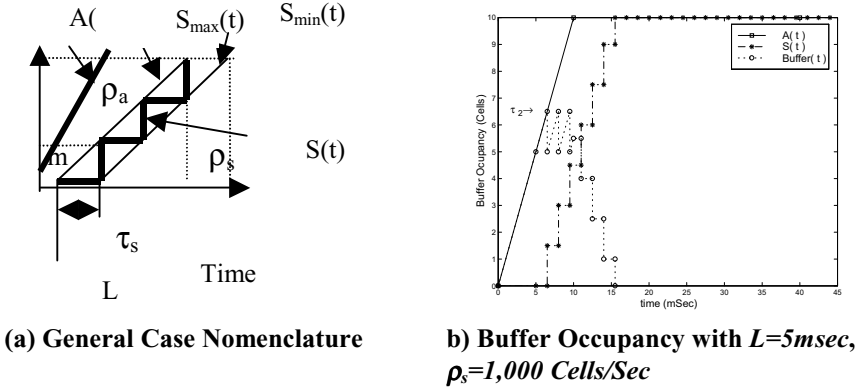


Fig. 2 Fluid Arrivals and Discrete Service

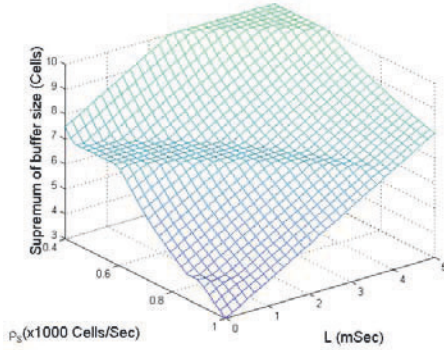


Fig. 3 Buffer Size as a Function of  $L$  and  $\rho_s$  for Worst-Case Fluid Arrivals and Discrete Service with service-latency

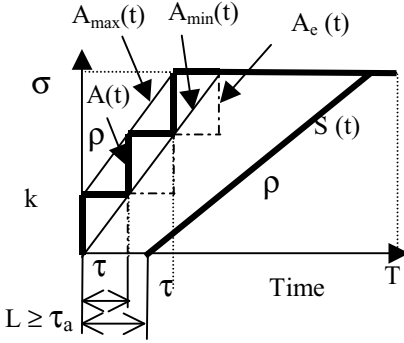
#### 4.4 Discrete Arrivals and Fluid Service

The main result in this section is equation (12) (refer to [12] for the proof), which presents a formula for the worst case buffer requirements when there is a discrete arrival function and a fluid service function. In brief, the intuition on locating the supremum from the fluid case is extended by whether the supremum occurs at the first step in the arrival function after  $t=L$ , i.e.  $t=\tau_I$ , or at the last step in the arrival function prior to  $t=\tau_\sigma$ , i.e.,  $t=\tau_d$ . The formulation is made more complex by consideration of

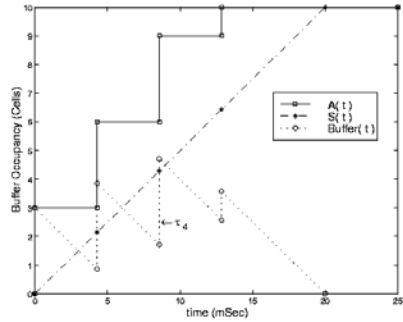
the arrival step size,  $k$ , in relation to the maximum burst size  $\sigma$ . Fig. 4 (a) illustrates the general case. In Fig. 4(b), a discrete arrival function is considered that has the following characteristics:  $\sigma=10$  Cells;  $\rho=400$  Cells/Sec;  $\rho_a=7$  Cells/Sec;  $k=3$  Cells. In Fig. 4(b) time is shown on the x-axis, and  $A(t)$  is shown as a solid line.  $S(t)$  is shown as alternating dash-dot line, and the maximum buffer occupancy is shown as a dotted line. In Fig. 4(b) the last step is less than  $k$ . The maximum buffer occupancy occurs at  $\tau_4$  (~9msec).

$$b \geq \begin{cases} \sigma & \text{if } \{L \geq \tau_\sigma\} & \text{or if } \{k \geq \sigma\} \\ \sigma - (\tau_\sigma - L)\rho_s & \text{if } c1 = \text{TRUE} & \text{and if } A(\tau_4) \geq \sigma - \tau_a\rho \\ A(\tau_4) - \rho_s(\tau_\sigma - L - \tau_a) & \text{if } c1 = \text{TRUE} & \text{and if } A(\tau_4) < \sigma - \tau_a\rho \\ k \left\lceil \frac{L}{\tau_a} \right\rceil & \text{if } c2 = \text{TRUE} & \text{and if } \left\{ k \leq \frac{\tau_1 - L}{\rho_s} \right\} \text{ and if } \tau_1 \neq \tau_\sigma \\ A(\tau_1) - S(\tau_1) & \text{if } c2 = \text{TRUE} & \text{and if } \left\{ k > \frac{\tau_1 - L}{\rho_s} \right\} \text{ and if } \tau_1 \neq \tau_\sigma \\ \max(A(L), (A(\tau_1) - S(\tau_1))) & \text{if } c2 = \text{TRUE} & \text{and if } \tau_1 = \tau_\sigma \end{cases} \quad (12)$$

where  $c1 = \{L < \tau_\sigma\} \text{ AND } \{\rho_a > \rho_s\}$   
 $c2 = \{L < \tau_\sigma\} \text{ AND } \{\rho_a \leq \rho_s\}$



(a)  $\tau_a < L < \tau_\sigma$ ,  $\rho_s > \rho_a$

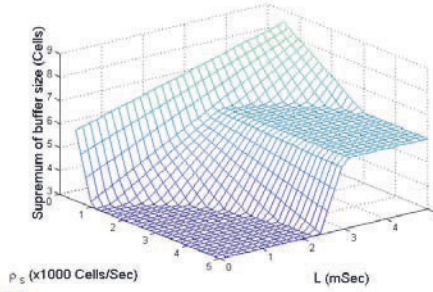


(b) Buffer Occupancy with  $L=0$  msec,  $\rho_s=500$ cells/s. The discrete arrival function used was:  $\sigma=10$  Cells;  $\rho=250$  Cells/S;  $\rho_a=1000$  Cells/S;  $k=3$  Cells.

**Fig. 4** Discrete Arrivals and Fluid Service

Fig. 5 illustrates the buffer size behavior as a function of  $L$  and  $\rho_s$ . Note that the minimum buffer size in Fig. 5 is not zero, but the  $k(=3$ cells) on the vertical axis. When  $\rho_s > \rho_a$ , the buffer size shows discrete steps due to the discrete arrivals. When  $\rho_s < \rho_a$ , the buffer size is a continuous function. Although this example uses a small value

of  $\sigma$  ( $\sim 10$  cells), for connections with large values of  $\sigma$ , the potential reduction in buffer requirements could be significant. This numerical example is also considering the effects on one connection. Aggregating the result over the number of connections supported at the interface provides for a potentially large reduction in buffer requirements. For example, if we can operate with a low latency of  $\sim 1\text{msec}$  and a  $\rho_s = 3,000 \text{ Cells/sec}$ , a 70% reduction in buffer requirements can still be achieved even after considering discrete arrival effects.



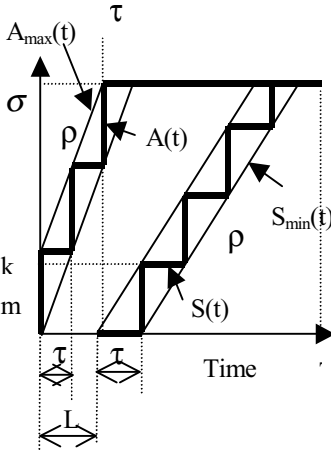
**Fig. 5** Buffer Occupancy Supremum as function of  $L$  and  $\rho_s$

#### 4.5 Combined Discrete Arrival and Departure Effects

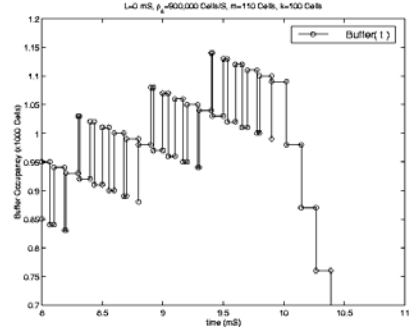
The effect of combining discrete arrivals and service is shown in Fig. 6(a). The intuition followed in the fluid cases to locate the supremum is not sufficient here. In Fig. 6(b), rather than showing the evolution of discrete  $A(t)$  and  $S(t)$  functions (as in e.g. Fig. 2(b)) we chose to show only the detail of the buffer occupancy for an example where the supremum does not occur at the location intuitively expected. Consider the case where the periods ( $\tau_a, \tau_s$ ) of the arrival and service curves are not equal. As the phase of the two discrete functions changes, eventually a point is reached where one discrete function steps twice between two steps in the other discrete function. This results in local maxima or minima in the buffer occupancy. This invalidates the previous intuition about where the supremum occurs.

Where both  $k$  and  $m$  are small with respect to  $\sigma$ , the bounding model considering discrete effects is appropriate. The fluid model bounding the discrete arrival function was derived as equation (5). The worst-case fluid model bounding the discrete service function was derived as equation (10). Combining these two leads to equation (13) (refer to [12] for the proof). If only one of  $k, m$  was small, then a bounding model could be derived based on the material in the previous sections. The supremum can be evaluated by exhaustively computing the buffer occupancy at each discrete step in the arrival or service functions. If neither  $k$  nor  $m$  is small, then there will be few steps making the numerical solution of the buffer supremum faster.

$$b \geq \begin{cases} \sigma & \text{if } \{(L + \tau_s) \geq \tau_\sigma\} \\ \sigma - (\tau_\sigma - L - \tau_s)\rho_s & \text{if } \{(L + \tau_s) < \tau_\sigma\} \text{ and } \{\rho_a > \rho_s\} \\ k + (L + \tau_s)\rho_a & \text{if } \{(L + \tau_s) < \tau_\sigma\} \text{ and } \{\rho_a \leq \rho_s\} \end{cases} \quad (13)$$



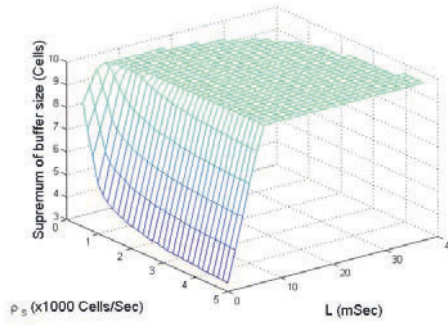
(a) General Nomenclature



(b) Buffer Occupancy Behavior Detail

**Fig. 6** Combining Discrete Arrivals and Service

Fig. 7 illustrates the bound on buffer size required after considering the discrete effects using equation (13). This is very similar in shape to Fig. 1, with the minimum possible buffer size now being  $k$  (3 Cells in this example) rather than zero as in Fig. 1. The position of the knee where buffer reduction below  $\sigma$  occurs is also moved slightly (to  $L \sim 8 \text{ msec}$ ) due to the additional service-latency introduced by discrete service effects. Despite the discrete effects, significant buffer reductions are still possible where the service-latency can be reduced below this knee. As in Fig. 1, when the latency is below the knee point, increasing the service rate to match the peak arrival rate provides potential buffer reductions, increasing beyond the peak arrival rate does not provide any additional reduction in buffer size. Although reduction in the number of cells in this example is small, the potential reduction is significant for bursty connections with large values of  $\sigma$ , and when considered aggregated over the potentially large numbers of connections on an interface.

**Fig. 7** Bound on Buffer Space Considering Discrete Effects with  $k=3$ ,  $m=2$

## 5 Buffer Size Reduction in WRR

In this section, we provide a numerical illustration of the feasibility and scale of buffer size reduction for a specific WRR scheduler. We assume a node with  $n$  identical input lines of speed  $C$  Cells/Sec. The maximum number of simultaneous arrivals is then one per link, i.e.  $k=n$  (for a MPt-Pt connection). The peak arrival rate is  $\rho_a = kC$ , and the maximum aggregate arrivals are still constrained to  $\sigma$ .  $\tau_a$  is given by Equation (14). The maximum number of arrivals in the worst-case-burst have been received by the time given in equation (3). Consider a scheduler that offers a rate guarantee of some fraction of the link bandwidth ( $C$ ) to a class  $i$ . This class is given a integer weight  $\phi_i$ . A simple interpretation is that the integer weights,  $\phi_i$ , correspond to the number of units of service to be provided in one round by the WRR server, where the units of the weights are the units of service (e.g. bits, cells, packets). We assume the step size for the discrete service is  $m = \phi_i$ . Then  $\rho_s$  is given by equation (15), and  $\tau_s$  is given by equation (16)

$$\tau_a = \frac{k}{\rho_a} = \frac{1}{C} \quad (14)$$

$$\rho_s = \frac{\phi_i}{\sum_{\forall j} \phi_j} C \quad (15)$$

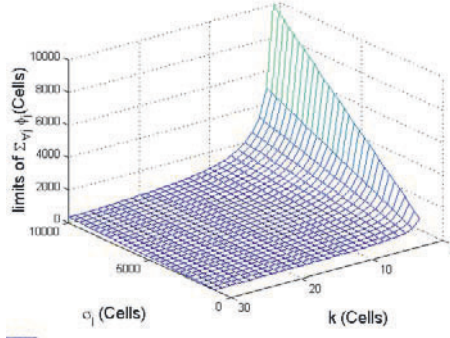
$$\tau_s = \frac{m}{\rho_s} = \frac{\sum_{\forall j} \phi_j}{C} \quad (16)$$

The worst-case service-latency ( $L$ ) is the service-latency due to the discrete arrival processes. i.e.  $L = \tau_a$ . The buffer space requirements (assuming  $k$  and  $m$  are small with respect to  $\sigma$ ) corresponding to this can then be derived by substituting in equation (13).

### 5.1 Feasible Region for Reduced Buffer Size

We assume  $k$  is a positive integer, and for all non-trivial cases  $C > \rho_s$  and hence  $\rho_a \geq \rho_s$ . From equation (13), we have the constraint on service starting before  $\tau_\sigma$ . This can be developed into a constraint on the maximum frame size as shown in equation (17). This constraint is illustrated in Fig. 8 where frame sizes that permit buffer reduction are in the region below the surface. The frame size constraint is linear in  $\sigma$ , but has an inflexion point as a function of  $k$ . In order to have a reasonably large frame size to accommodate connections with large  $\sigma$ , we need to keep  $k$  small, e.g.  $k < 10$ .

$$\text{Maximum frame size} = \sum_{\forall j} \phi_j < \frac{\sigma_i - k}{k} - 1 \quad (17)$$



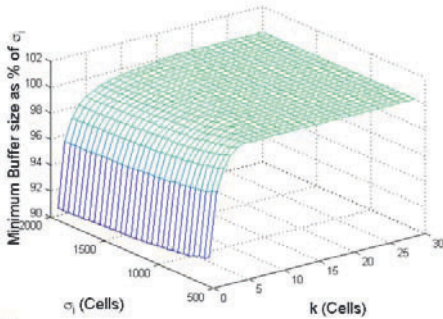
**Fig. 8** maximum frame size ( $\sum_{\forall j} \phi_j$ ) as a function of  $k$  and  $\sigma$

## 5.2 Reduced Buffer Size

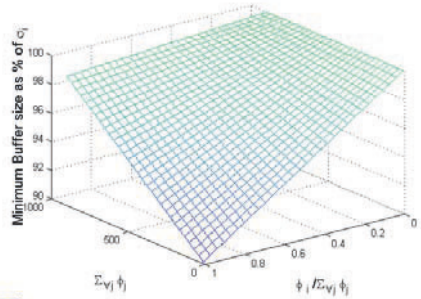
Equation (13) can be reduced in this region of interest as equation (18).

$$b_i \geq \sigma_i - \left( \frac{\sigma_i - k}{k} - 1 - \sum_{\forall j} \phi_j \right) \frac{\phi_i}{\sum_{\forall j} \phi_j} \quad (18)$$

Fig. 9 illustrates the behavior of equation (18). To avoid worst case buffer overruns, the buffer should be provisioned above this surface. Significant reductions in buffer size are only possible for small values of  $k$  e.g. systems with small fan-in. Although there is some sensitivity to the burst size,  $\sigma_i$ , at large values of the burst size  $\sigma_i$ , even a small percentage reduction can represent a large amount of memory. This example shows buffer reductions of up to 10% from the burst size  $\sigma_i$ , but further reductions are also possible by considering the effects of the weights used.



(a) as function of  $k$  and  $\sigma$  for the case of  $\phi_i = 10$ ,  $\sum_{\forall j} \phi_j = 100$ .



(b) as functions of the connection weights for the case of  $k=10$  Cells and  $\sigma_i=10,000$  Cells

**Fig. 9** Buffer Size Requirement per Inequality (18)

Buffer size reductions from  $\sigma_i$  are clearly possible. However this appears to only be significant when the proportional weight allocated to the over-allocating connection is close to 1 and the maximum frame size ( $\sum_{\forall j} \phi_j$ ) is small.

## 6 Buffer Sizing Conclusions

In this paper, more precise (than fluid model) formulations for buffer size requirements have been developed to consider the effects of worst-case discrete arrival and discrete service, as well as service latency. When both  $m$  and  $k$  are small compared to  $\sigma$  the formulation for the combined case of discrete arrivals and discrete service (see equation (13)) provides a suitably tractable bound on the worst-case. The worst-case formulations for discrete arrivals and fluid service (see equation (12)) or fluid arrivals and discrete service (see equation (11)) are exact solutions. These formulations should be used for buffer size calculations when the discrete nature of either the arrival or service function must be considered (i.e. when one of  $m$  or  $k$  is not small compared to  $\sigma$ ). When both  $m$  and  $k$  are not small compared to  $\sigma$  then the evolution of the buffer occupancy needs to be evaluated in more detail. The computational complexity of determining the supremum of the buffer occupancy reduces to evaluating the buffer occupancy at each step in the arrival and service functions. With both  $m$  and  $k$  large, there will be few steps between the onset of service and the completion of the arrival burst and an exhaustive evaluation is feasible in a reasonable time. The numerical application of equation (13) to the selection of weights for a discrete WRR scheduler has been provided as an illustration. In this example, buffer size reduction below  $\sigma$  is possible, (up to 10% in Fig. 9) but the gains are most significant when the sum of all the weights is small, the burst size,  $\sigma$  is large, and the fan-in,  $k$ , is small. In these cases, even a small percentage reduction (say 10%) in the buffer requirements can result in a significant reduction in the number of cells that must be stored. For network elements requiring high-speed memory buffers, the cost savings can be significant.

## References

1. Agrawal, R., R. Cruz, C. Okino, R. Rajan, "Performance Bounds for Flow Control Protocols", IEEE/ACM Transactions on Networking, Vol. 7, No. 3, June 1999.
2. Chang, C-S, "On Deterministic Traffic Regulation and Service Guarantees: A Systematic Approach by Filtering", IEEE Transactions on Information Theory, Vol. 44, No. 3, May 1998, pp1097-1110.
3. Elwalid, A., D. Mitra, R.H. Wentworth, "A New Approach for Allocating Buffers and Bandwidth to Heterogeneous Regulated Traffic in an ATM Node", IEEE Journal on Selected Areas in Communications, Vol 13, No. 6, August 1995, pp 1115-1127.
4. Garg, R., X. Chen, "RRR: Recursive Round Robin Scheduler", IEEE Globecom'98, paper S94-4, November 1998.



5. Guerin,R., S.Kamat, V.Peris, R.Rajan, "Scalable QoS Provision through Buffer Management", Computer Communications Review, Vol. 28, No. 4, 1998
6. Katevenis, M., S. Sidiropoulos, C. Courcoubetis, "Weighted Round Robin cell Multiplexing in a General Purpose ATM Switch Chip", IEEE Journal on Selected Areas in Communications, Vol. 9, No. 8, October 1991.
7. LeBoudec, J-Y, "Application of Network Calculus to Guaranteed Service Networks", IEEE Transactions on Information Theory, Vol. 44, No. 3, May 1998, pp1087-1096
8. LoPresti, F., Z-L.Zhang, J.Kurose, D.Towsley, "Source Timescale and Optimal Buffer/Bandwidth Tradeoff for Regulated Traffic in an ATM Node", Proceedings of IEEE INFOCOM'97 .
9. Sasaki,G., "Input Buffer Requirements for Round Robin Polling Systems", Performance Evaluation, Vol. 18, 1993, pp 237-261.
10. Stiliadis, D., A. Varma, "Latency-Rate Servers: A General Model for Analysis of Traffic Scheduling Algorithms", IEEE Infocom'96, San Francisco, March 1996, pp 111-119.
11. Wright, S., Y. Viniotis, "Simulation Study of Maximal Delays in Various Link Scheduling Algorithms", IEEE Globecom'98, paper S117-7, November 1998.
12. Wright, S., "Delay Bounds and Connection Admission Control for Burstiness-Class Queueing", Ph.D. Thesis, North Carolina State University, 1999.
13. Wright, S., Y. Viniotis, "ATM Network Procedures Supporting Delay QoS", IEEE/IEICE ATM Workshop 99, Kochi City, Kochi, Japan May 24-27, 1999.
14. Wright, S., Y.Viniotis, "Burstiness-Class Based Queueing in ATM Networks Supporting Delay QoS Bounds", IEEE Infocom 2000, Tel Aviv, Israel, March 2000.

# Distributed Input and Deflection Routing Based Packet Switch Using Shuffle Pattern Network

Thai Thach Bao, Hiroaki Morino, Hitoshi Aida, and Tadao Saito

Department of Information and Communication Engineering, Faculty of Engineering,  
University of Tokyo,  
7-3-1 Hongo Bunkyo-ku Tokyo 113-8656 Japan

**Abstract.** In this paper, a scalable variable-length packet switch using multistage interconnection network is proposed. The architecture consists of multiple non-buffer switch elements, interconnected with perfect shuffle pattern and with ring topology. Each input port is connected to a switch element in a different stage, and packets applied from these ports are routed to their destined output ports based on principle of deflection routing. The proposed switch has following features; (1) To achieve certain packet loss rate, required number of stage to achieve certain packet loss rate is less than the same type of switches due to efficient use of switch elements. (2) Packet loss rate is almost unchanged regardless of traffic patterns. In performance evaluation, two types of non-uniform traffic are given and it is shown that the switch can achieve lower packet loss rate than existing deflection routing based switch.

## 1 Introduction

In recent years, demands of voice, data, video traffic increase explosively due to the success of the Internet. Corresponding to this situation, required capacity for backbone packet switch in the future will reach Tera bit/sec, one thousand times of today's packet switch capacity. Furthermore, traffic demand will be concentrated on large size file transfer and broadband video communication, and it is expected for the switching system to cope with these demands.

From this viewpoint, we have been investigating a new framework of multimedia network that is hybrid system of variable-length packet switch and TDM based switch in place of solution with ATM switch[1]. In this system, non real-time traffic such as data transfer traffic is transmitted via variable-length packet switch, while real-time traffics such as voice, video is transmitted via TDM based switch. Access network is constructed using integrated TDM technology, that provides unified interface to users. According to this framework, we have examined variable-length packet switch architecture having capability to handle traffic of Tera bit/sec. In order to design high capacity packet switch of this level, conventional shared bus architecture is insufficient while parallel switching using crosspoint switch is indispensable. Implementations of crosspoint switch are roughly classified into two categories; input buffering switch and output buffering switch.

Input buffering switch have advantage of simplicity of switch elements and buffer management and high throughput is achieved by combination of virtual output queuing and efficient scheduling[2]. However, maximum switch size will be limited by processing time of scheduling, and this problem will be more serious when port speed is increased.

For the purpose of implementation of large switch up to about  $100 \times 100$ , output buffering switch is noticeable due to its simple switching operation. The feature that it requires amount of wiring and pins is not a big problem with the recent progress of VLSI technology. Among several output queuing switches, authors have investigated deflection routing based switches, which are suitable for variable-length packet switch. For a deflection routing based switch in ATM, several methods have been proposed such as Tandem Banyan switch[3] or MS4[4] switch, but it is not optimized for variable-length packet switch.

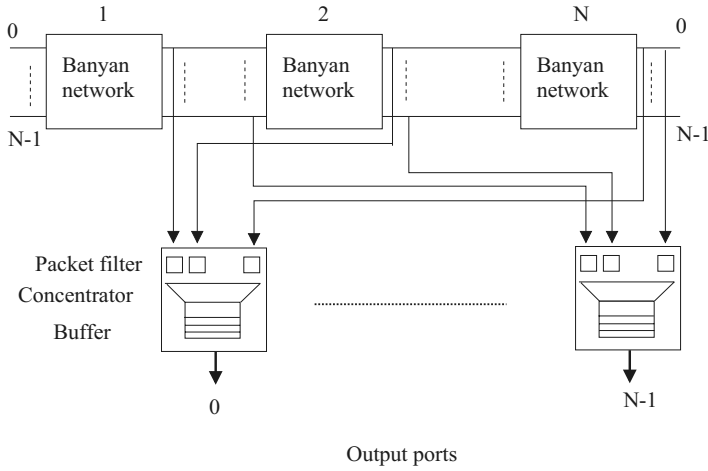
In this paper, a deflection routing variable-length packet switch based on input port distribution is proposed. This switch consists of switching elements interconnected through shuffle pattern[5], and the switch exploits ring topology. Positions of input ports are distributed according to the rotation direction within the ring. Therefore, traffic applied to the switch is distributed over the switch, and utilization of switch elements is improved. As a result, required number of switch stages is less than conventional switches in order to achieve certain packet loss rate.

This paper is organized as follows. In Section 2, features of conventional type of deflection routing based switch are described, and further research issues are presented. In Section 3, architecture and operation of the proposed switch is described. In Section 4, results of performance evaluation are shown and it is indicated that performance of the switch does not change under both uniform and non-uniform traffic pattern. Finally, conclusion and future work are presented in Section 5.

## 2 Background of Research

In Figure 1 and Figure 2, Tandem Banyan switch and Multi Single-Stage-Shuffling Switch(MS4) are presented as typical deflection-routing based switches. Though these are originally designed for fix-length ATM cell switch, they can be easily applied to variable-length packet switch. In these switches, when internal packet conflict occurs at some switch element, one packet is properly routed to destined port and others are misrouted to the rest of ports. In Tandem Banyan switch, misrouted packets will start routing again from a succeeding Banyan plain. On the other hand, they are allowed to restart routing from succeeding stage in MS4 switch. Comparing performance of two switches, MS4 switch will be better since it can utilize switch element more efficiently.

In both these switches, however, all applied packets enter the switch at the first stage. The utilization efficiency of switch elements in preceding stages is higher than that in latter stages resulting in a switch resources waste. As a result, probability of packet conflict in a switch element is high when the element



**Fig. 1.** Tandem-Banyan switch

is close to the first stage. If applied traffic load can be distributed equally over each stage of the switch, the number of stages required to achieve certain packet loss rate will decrease.

As an approach to distributing traffic within switch, Ring Banyan switch[6] was proposed, which interconnects switch elements in ring topology. Ring Banyan is shown in Figure 3. The routing operation is the same as Tandem Banyan switch, and applied traffic is distributed uniformly within the network. It is shown that required number of stage is less than that of Tandem Banyan switch.

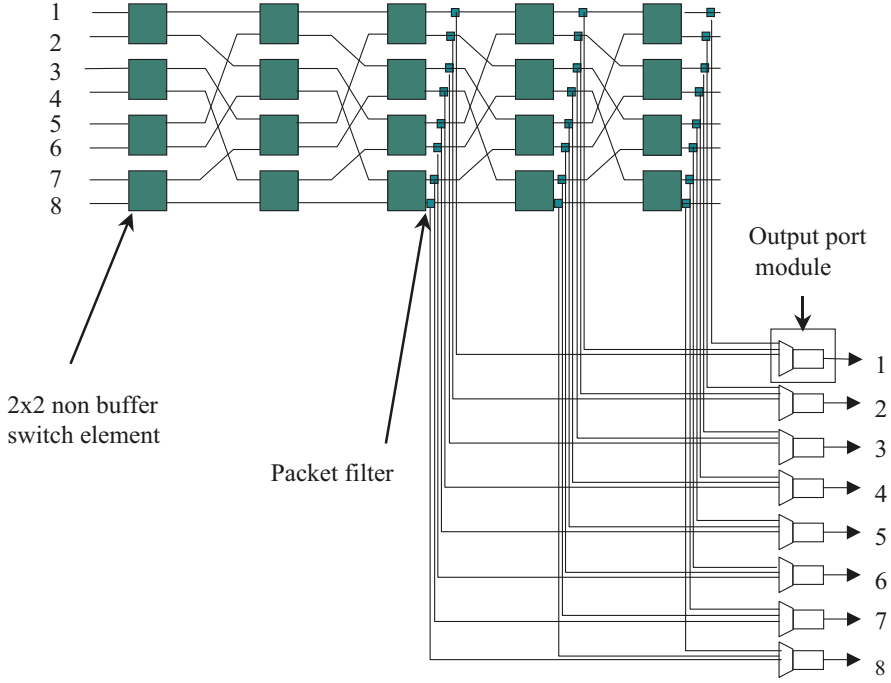
In Ring Banyan switch, however, misrouted packets waste switch elements until they enter next banyan plain. In variable-length packet switching, they may prevent other packets from being routed properly, which lead increase of packet loss. This type of loss can be reduced if misrouted packets can restart routing from the next stage.

### 3 Proposal of Distributed Input Shuffle Pattern Switch with Deflection Routing

In order to resolve problems in Ring Banyan switch, a new type of switch with ring topology is proposed, which optimizes the interconnection pattern of switch elements.

#### 3.1 Basic Architecture and Operation

The proposed switch is shown in Figure 4, where switch elements are connected through shuffle pattern. An applied packet is routed based on principle of self



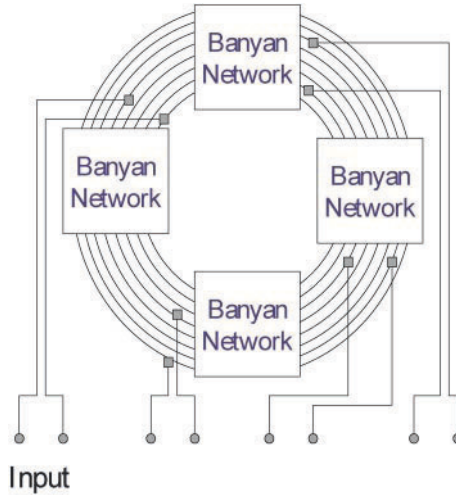
**Fig. 2.** Deflection routing based shuffle pattern switch(MS4 switch)

routing and rerouting. When a packet reaches a packet filter connected to the destination output port, it leaves the switch.

According to the ring topology, an applied packet can pass through arbitrary large number of stages. Therefore, a new packet may conflict with a packet coming from the previous stage. To prevent this problem, some buffer is provided at input port as shown in Figure 6. The new applied packet and a packet from the previous stage are stored in this buffer, which operates on first-comes-first-served basis. At each output module, a concentrator is provided to reduce the number of output buffers. Structure of concentrator and buffer management is the same as those in Knockout switch[7].

Before an operation of the switch is described in detail, several notations are given.  $N$  represents the number of input/output ports.  $n = \log_2 N$  represents the number of stages in an ordinary Banyan network.  $K$  is the number of stages in the switch. The local destination address of a packet will be represented in binary bits as  $(d_1 d_2 \dots d_n)$  (where  $d_1$  is the most significant bit). Proposed switch  $(N \times N)$  is constructed using  $K (\geq n)$  stages, each stage includes  $N/2$  rows of  $2 \times 2$  non-buffer switch elements.

Before a packet is applied to the network, the local header is generated and attached in front of each packet. The local header of each packet includes a des-



**Fig. 3.** Ring Banyan switch

tion address field, which contains the destination port represented  $d_1d_2\dots d_n$ , and a counter field  $C$ , which is initially set to  $n$ . When a packet is applied to a switch element, output port of the element to which the packet is transmitted is decided according to the corresponding bit in the destination address field. The switch element refers the  $d_{n-C+1}$  bit of the field, and the packet is transmitted to the upper port if it is zero, and to the lower port if it is one.

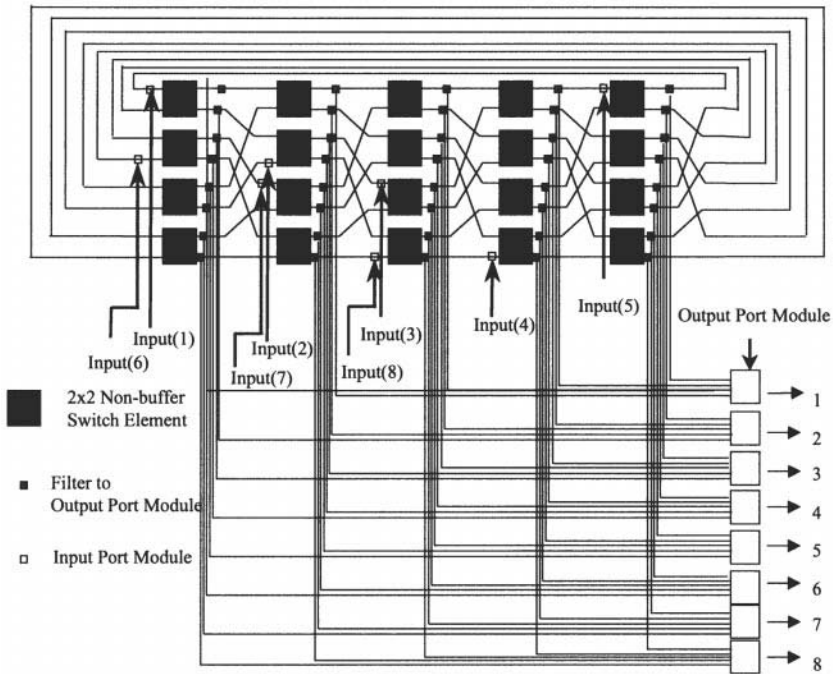
When the packet is routed to the port successfully, the counter is decremented. Otherwise, when the port is occupied by another packet, the packet is routed to the other port, and the counter is set to  $n = \log_2 N$ , and it continues routing from the next stage. If two packets are applied at the same time and they require the same output port of the switch element, the packet closest to its destination (smaller value of counter field) is served as the highest priority.

In this way, the counter of a packet is decremented whenever the packet is successfully routed to the destined output port of a switch element, and the packet reaches packet filter connected to its destination output port of the switch when the counter is zero. In other words, a packet can reach its destination port of the switch fabric if and only if it successfully passes through  $n$  consecutive stages.

The function of packet filters at the output ports of each stage is to examine the content of the counter of each packet and to transmit packet to the following stage for continuing routing or to output port module.

## 4 Performance Evaluation

In this paper, it is assumed that variable-length packet switch of Tera bit/sec in the future has OC-12 (622.08 Mega bit/sec) interface and its size is about



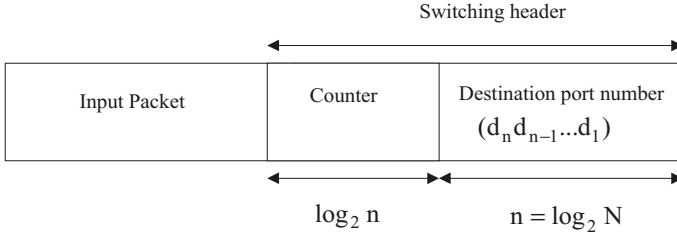
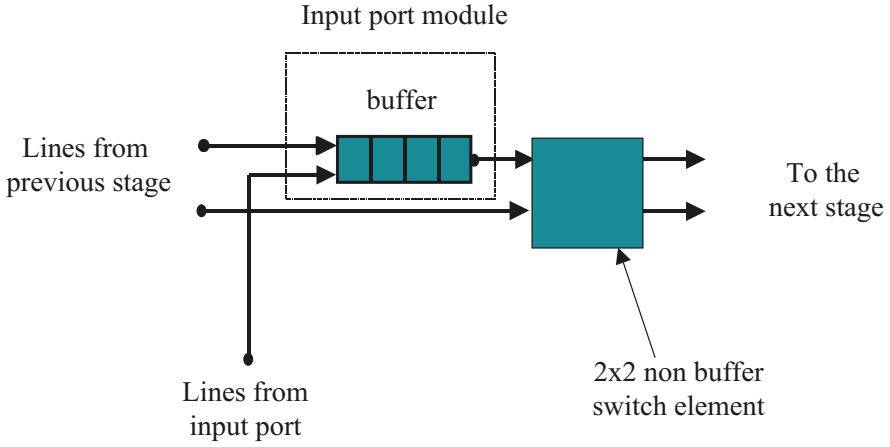
**Fig. 4.** Proposed switch

4000 x 4000. Considering today's VLSI technology, this size of switch will be constructed by two-stage interconnection of 64 x 64 switch implemented in a chip. On this assumption, size of the switch evaluated here is also set to 64 x 64.

In most cases, the switch performance strongly depends on patterns of applied traffic. In this section, two traffic conditions are prepared where destined output ports of applied packets are distributed in uniform and non-uniform pattern. Under these conditions, packet loss rate of the proposed switch is evaluated by computer simulation.

In our simulations, size of packets varies uniformly from 20 bytes to 1500 bytes according to the IP packet format, and time interval of packet arrival follows negative exponential distribution.

Packet loss shown in the following figures includes loss in the shuffle switch fabric and loss in the input modules, and it does not include loss in the output modules. It is because loss in output port module is approximately the same as other output buffering switches.

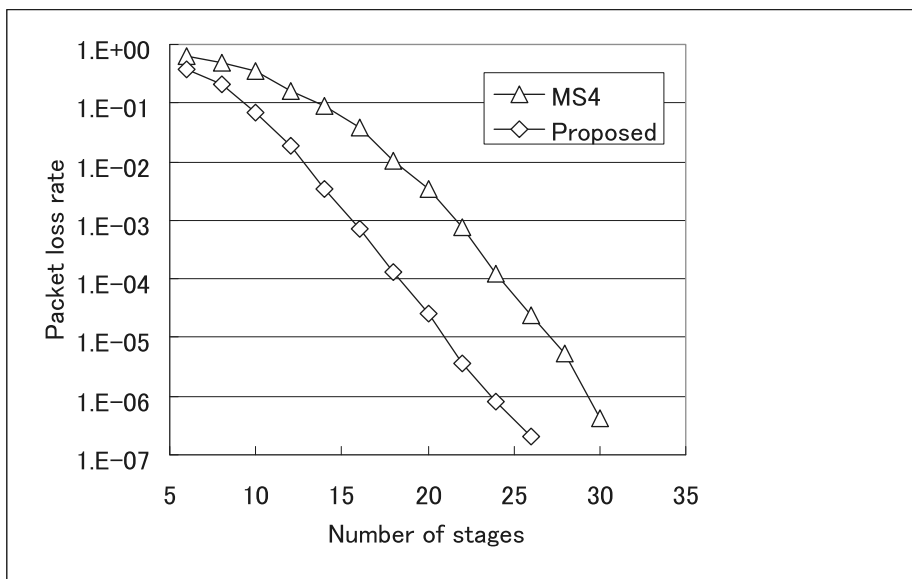
**Fig. 5.** Packet format**Fig. 6.** Input module structure

#### 4.1 Packet Loss Rate under Uniform Traffic Pattern

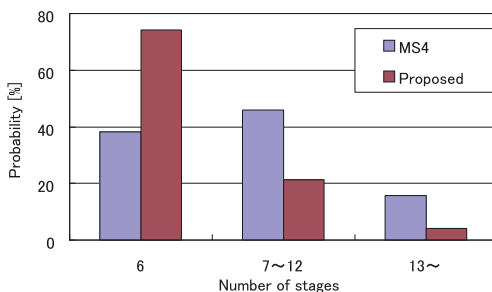
In this condition, the destination output port of a packet is uniformly distributed among all output ports.

Figure 7 plots the packet loss rate in the switch fabric as a function of  $K$  for  $N = 64$  at load  $p = 0.6$ . The packet loss rate of the MS4 switch with the same traffic and switch conditions is also plotted for comparison. For example, a packet loss rate of  $10^{-6}$  is achieved with  $K = 24$  in the proposed switch while the MS4 switch needs 29 stages. Figure 8 illustrates probability distribution of required number stages by which a packets reaches the destination output port in the simulation. Under this condition, required number of stages is six in both switches when the packets do not suffer any conflict. The figure shows that probability that a packet arrives at the destination port by six stages in the proposed switch is 74%, while it is 38 % in MS4 switch. Therefore, it can be said that average transmission delay of packets in the proposed switch is less than





**Fig. 7.** Packet loss rate in the switch fabric under the uniform traffic



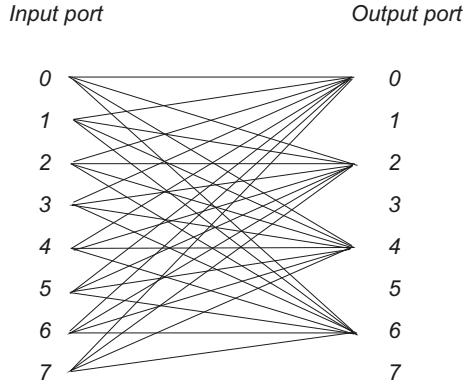
**Fig. 8.** Probability distribution of required number of stages by which a packet reaches its destination port

that of MS4 switch. This result shows that technique of distributing input port has great effectiveness in reducing probability of conflict within the network.

## 4.2 Packet Loss Rate under Non-uniform Traffic Pattern

All the non-uniform traffic patterns studied here are considered to satisfy properties shown below[8]:

- packets arrive at all input ports with the same probability  $p$



**Fig. 9.** Rectangular non-uniform traffic pattern (An example for 8 x 8 switch)

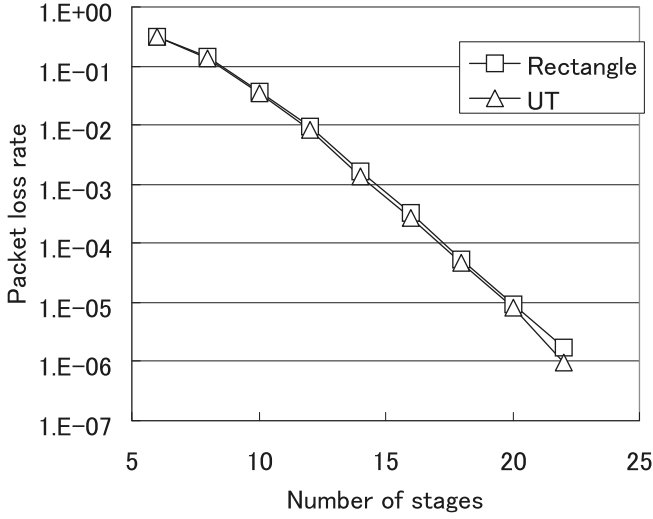
- arriving packets request a destination port according to a non-uniform distribution.

For the destined output port distribution, rectangular traffic pattern and hot-spot traffic pattern are provided.

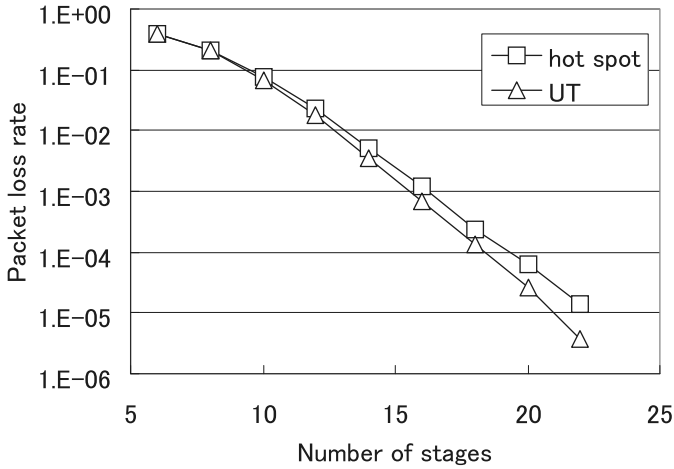
**Rectangular Traffic Pattern** In this traffic pattern, all arriving packets are addressed to only a subset of all output ports. We consider the case that number of the ports is  $L = N/2$  and the distance between the ports is the furthest. An example of the pattern for 8 x 8 switch is shown in Figure 9. Figure 10 shows the packet loss rate at load  $p = 0.5$  as a function of  $K$ , where each of rectangular traffic pattern and uniform traffic pattern (UT) is offered. Though larger number of sta ges  $K$  required in rectangular pattern to achieve the same loss rate as UT, the difference can be ignored.

**Hot Spot Traffic** Next, the performance under hot spot traffic is studied. A hot-spot traffic is defined as a traffic pattern in which a single hot-spot of high access percentage is superimposed on a background of UT[9]. Supposed that  $h$  is the rate of packets addressed at the hot-spot, we consider the case of  $h = 10\%$  and emulation results are plotted in Figure 11. The figure shows that effect of hot spot traffic on increase of loss rate is quite small.

The above results attained by simulations under various non-uniform traffic patterns show that the distributed input shuffle switch not only achieves high performance under uniform traffic, but also it is robust under non-uniform traffic pattern which may rise in many real applications.



**Fig. 10.** Packet loss rate in the switch fabric under the rectangular traffic



**Fig. 11.** Packet loss rate in the switch fabric where hot-spot traffic is superimposed

## 5 Conclusion

In this paper, we have introduced a deflection routing based packet switch using shuffle pattern network and technique of input ports distribution. By distributing

input ports over all stages of the switch, probability of internal packet conflict is reduced, and low packet loss rate is achieved. Moreover, computer simulations show that the switch is robust under non-uniform traffic pattern.

In a future work, several technical requirements for actual implementation of the switch will be investigated.

## Acknowledgement

The authors would like to acknowledge the support of JSPS(JSPS-RFTF96P00601).

## References

1. T.Aoki, "Post-ATM Network Architecture", Ph.D. Thesis, The University of Tokyo, Dec. 1997.
2. Nick McKeown, Martin Izzard, Adisak Mekkittikul, Bill Ellersick and Mark Horowitz, "The Tiny Tera: A Packet Switch Core", IEEE Micro Jan/Feb 1997, pp. 26-33.
3. F.A.Tobagi, T.Kwok and F.M.Chiussi, "Architecture, Performance, and Implementation of the Tandem Banyan Fast Packet Switch", IEEE Journal on Selected Areas in Communications, Vol.9, No.8, Oct. 1991, pp.1173-1193.
4. Ra'ed Y Awdeh and H T Mouftah, "MS4-a high performance output buffering ATM switch", Computer communications, Vol.18, No.9, Sep. 1995, pp.631-644.
5. H.S.Stone, "Parallel processing with the Perfect Shuffle", IEEE Transactions on Computers, Vol.C-20, No.2, Feb. 1971, pp.153-161.
6. T.Aoki, H.Aida, T.Saito, "A multi stage interconnection switch for high capacity IP switching", Proc. of The 55th IPSJ conference Sep. 1997. (In Japanese).
7. Y.Yeh, M.G.Hluchyj, and A.S.Acampora, "The knockout switch: A simple, modular architecture for high-performance packet switching", IEEE Journal on Selected Areas in Communications, Vol.SAC-5, No.8, Oct. 1987, pp.1274-1283.
8. S. Bassi, M. Decina, A. Pattavina, "Performance analysis of the ATM Shuffleout switching architecture under non-uniform traffic patterns", *Proc. of IEEE INFOCOM*, Florence, Italy 1992, pp.734-742.
9. G.F.Pfister and V.A.Norton, "Hot spot contention and combining in multistage interconnection networks", IEEE Transactions on Computers, Vol.C-34, No.10, Oct. 1985, pp.943-948.

# A Distributed Dynamic Scheduling Algorithm for a Terabit Multicast Packet Switch

Feihong Chen, Necdet Uzun and Ali N. Akansu

New Jersey Center for Multimedia Research  
Department of Electrical and Computer Engineering  
New Jersey Institute of Technology  
University Heights, Newark, NJ 07102  
Email: {fxc0407, uzun}@oak.njit.edu, ali@megahertz.njit.edu

**Abstract.** In this paper, we present a novel switch design of a large scale multicast packet switch which is featured by a modular switch architecture and a distributed resource allocation algorithm. Switch inputs and outputs are grouped into small modules called Input Shared Blocks (ISBs) and Output Shared blocks (OSBs). Input link sharing and output link sharing are cooperated intelligently so that no speedup is necessary in central switch fabric (ATMCSF). Cell delivery is based on link reservation in every ISB. Dual round robin rings connect ISBs to provide a fast and fair link resource allocation among ISBs according to a Queue Occupancy Based Dynamic Link Reservation (QOBDLR) algorithm. QOBDLR is a distributed algorithm in which an ISB can dynamically increase/decrease its link reservation for a specific OSB according to its local available information. Arbitration complexity is  $O(1)$ . Switch performance is evaluated through simulations for an  $256 \times 256$  switch. It is demonstrated that the proposed switch can achieve a comparable performance as the output queued switch under any traffic pattern.

## 1 Introduction

As the demanding bandwidth of Internet services continues to increase, switches and routers face challenging high capacity requirements. Many services, such as teleconferencing and entertainment video, are characterized by point-to-multipoint communication. As a promising candidate of the backbone core switching system for Broadband networks, ATM switches need to be scalable, cost-effective, and support multicasting. In this paper, we propose a novel switch design of a scalable multicast packet switch.

Switches and routers employing output queueing (OQ) proved to maximize throughput and optimize latency. These characteristics are very important for high throughput and supporting Quality of Service (QoS). However, output queued switches are limited by the bandwidth of commercially available memories, since they require to store incoming data from  $N$  inputs, i.e.,  $N$  times increase in the memory speed compared to an input buffered switch. Currently, it is practical to implement an output queued switch or router with an aggregated bandwidth of several 10Gb/s. However, it is impractical to build an output queued switch with a large number of ports and fast line rate.

On the other hand, input queued (IQ) switches become more attractive because switch fabric and input memory only need to run as fast as line rate. To

overcome head of line (HOL) blocking, virtual output queues are applied at every switch input together with some weighted input-output matching algorithms to achieve 100% maximized throughput [5][6][7][8]. But, most scheduling algorithms proposed for IQ switches use centralized schedulers, which collect traffic information from  $N$  switch inputs in every cell slot and need multiple iteration to determine the final input-output matching. Scheduling complexity of at least  $O(N^{2.5})$  becomes a main obstacle when a switch grows to a large size and has a very fast line rate. Situation can become even worse under multicast traffic.

In our previous work [9], we proposed a design of a large scale ATM switch using input and output link sharing. Switch inputs and outputs are grouped into small modules called Input Shared Block (ISB) and Output Shared Block(OSB). Under uniformly distributed input traffic, link sharing with round robin cell scheduling resolves output contention and eliminates the speedup requirement for central switch fabric. Switch leads to a comparable performance as the output queued switch under uniform multicast traffic. However, isolated ISBs prevent switch from achieving high performance under non-uniform multicast traffic.

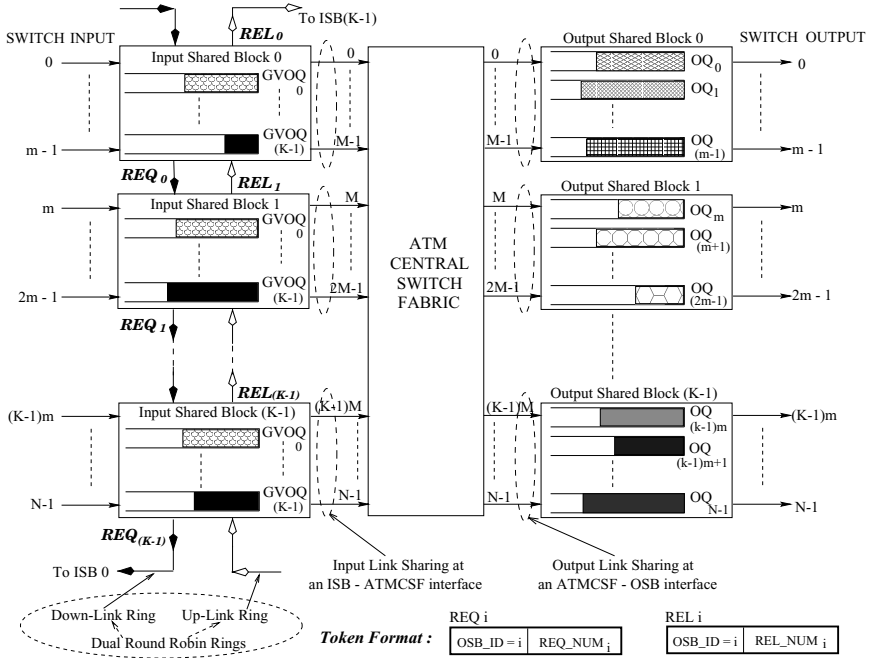
To support non-uniform traffic, in this paper, we propose an enhanced switch design in which ISBs are connected by dual round robin rings. Link request tokens (REQs) and link release tokens (RELs) circulate on the dual rings and pass ISBs one by one in a round robin manner. Every ISB should make link reservation in advance in order to obtain the desired links to the targeted OSBs. Cell delivery in every cell slot is based on link reservation in each ISB. We propose a Queue Occupancy Based Dynamic Link Reservation (QOBDLR) algorithm, in which every ISB can dynamically increase/decrease its link reservation for an OSB by “borrowing” and/or “lending” links from each other through REQ and REL tokens. QOBDLR is a distributed algorithm in the sense that an ISB modifies its link reservation according to its own local available information. Arbitration complexity is only  $O(1)$ . Hence, QOBDLR can achieve a fast and fair link resource allocation among ISBs. Performance evaluation demonstrates that the proposed switch can achieve a comparable performance as the OQ switch under any traffic pattern. The proposed switch can be easily extended to a large scale up to Terabits capacity.

This paper is organized as follows. In Section 2, we introduce a novel switch architecture using link sharing and dual round robin rings. In Section 3, we describe cell delivery and present the Queue Occupancy Based Dynamic Link Reservation (QOBDLR) algorithm. In Section 4, switch performance is evaluated with simulations for 256x256 switch. Conclusion is drawn in Section 5.

## 2 Switch Architecture

Fig 1 depicts the modular switch architecture. The proposed switch consists of four major components : *Input Shared Block (ISB)*, *Output Shared Block (OSB)*, *ATM Central Switch Fabric (ATMCSF)*, and *Dual Round Robin Rings*. Switch inputs and outputs are respectively grouped into  $K$  ISBs and  $K$  OSBs, where  $K = \frac{N}{m}$ . ISBs are connected by dual rings on which  $K$  link request tokens (REQs) and  $K$  link release tokens (RELs) circulate in a round robin manner. At every ISB-ATMCSF interface, there are  $M$  *input links* shared by  $m$  related switch inputs. At every ATMCSF-OSB interface, there are  $M$  *output links* shared by  $m$  grouped switch outputs. In this paper, we only consider the case of  $m = M$ ,

and the study of  $M > m$  which implies a virtual speedup in ATMCSF is the subject of our ongoing work. Applying input link sharing and output link sharing together is the unique feature of the proposed switch. Link sharing eliminates speedup requirement in ATMCSF.



**Fig. 1.** The proposed modular switch architecture : an  $N \times N$  switch consists of  $K$  ISBs,  $K$  OSBs, ATMCSF, and dual round robin rings;  $K = \frac{N}{m}$  and  $m = M$  in this paper.

## 2.1 Input Shared Block

An ISB can be a shared memory receiving multicast cells<sup>1</sup> from  $m$  ( $= M$  in this paper) related switch inputs. A multicast cell is saved once in an ISB instead of keeping  $j$  identical cell copies (assume,  $j$  is the fanout of a multicast cell,  $1 \leq j \leq N$ ). We propose a *Grouped Virtual Output Queue (GVOQ)* scheme for shared memory management in an ISB.

As shown in Fig 1, every ISB only maintains  $K$  ( $= \frac{N}{m}$ ) grouped virtual output queues. Each grouped virtual output queue is a linked list of the multicast cells targeting a same OSB. If a multicast cell has more than one destination to an OSB, only a single connection carrying all desired destinations is attached to the related grouped virtual queue. Hence, a cell delivered from an ISB may carry multiple destinations, then will be stored into every related output queues when

<sup>1</sup> A multicast cell may have one or multiple destinations. Hence, multicast traffic includes unicast traffic.

the cell is received by an OSB. GVOQ follows FIFO principle to receive cells from switch inputs and deliver cells to ATMCSF.

Since an ISB-ATMCSF interface has a capacity of  $M$  links, an ISB can deliver at most  $M$  cells to ATMCSF in every cell slot. An ISB can send a cell through any of the  $M$  shared links. Input link sharing is able to avoid link starvation when some GVOQ is empty, because other GVOQs can utilize the idle link to deliver their cells. Input link sharing results in an improved performance.

## 2.2 Output Shared Block

As shown in Fig 1, an OSB is a shared memory containing  $m$  ( $= M$  in this paper) output queues. In every cell slot, each output queue delivers one cell out of the related switch output. Since an ATMCSF-OSB interface only supports  $M$  links, an OSB can accept at most  $M$  cells from the central switch fabric in every cell slot. ATMCSF can use any of the  $M$  shared links to transmit a cell to an OSB. Without output link sharing, if more than one cell goes to a same switch output, either cells are blocked, or it is necessary for the switch fabric to speedup. However, output link sharing can avoid both problems.

## 2.3 Central Switch Fabric

The central switch fabric should keep the same cell sequence for those cells which are delivered from an ISB to a same OSB. Apart from this, no other restrictions are placed on the central switch fabric. It can be any type of switch fabric (for example Abacus switch [4]), and no speedup is necessary because of input link sharing and output link sharing.

## 2.4 Dual Round Robin Rings

ISBs are connected by dual rings : a down-ward ring conveys link request tokens (REQs); and an up-ward ring carries link release tokens (RELs). At any time, there are  $K$  REQ tokens and  $K$  REL tokens circulating on the dual rings respectively and passing ISBs one by one in a round robin manner. Each OSB (e.g. the  $i^{th}$  OSB) is correlated with a REQ token (e.g.  $REQ_i$ ) and a REL token (e.g.  $REL_i$ ).

Both REQ token and REL token contain two fields (shown in Fig 1) : (1) “OSB\_ID” is the identification of an OSB; (2) “REQ\_NUM” indicates how many link requests are issued for the identified OSB. Or, “REL\_NUM” records the number of released links which are available to be reserved at the related ATMCSF-OSB interface.

# 3 Cell Scheduling

## 3.1 Cell Delivery

Cell delivery is based on link reservation in every ISB. Each ISB has a *link reservation vector* and a *queue occupancy vector*. We use  $LK\_RSV^i$  and  $Q^i$  to represent the two vectors in the  $i^{th}$  ISB ( $0 \leq i, j < K$ ) :



- $\text{LK\_RSV}^i = [r_0^i, r_1^i, \dots, r_{(K-1)}^i]$ ; Link Reservation Vector in the  $i^{\text{th}}$  ISB. Where  $r_j^i$  indicates how many links at the ATMCSF-OSB  $j$  interface are reserved by the  $i^{\text{th}}$  ISB,  $0 \leq r_j^i \leq M$ .

- $\mathbf{Q}^i = [q_0^i, q_1^i, \dots, q_{(K-1)}^i]$ ; Queue Occupancy Vector in the  $i^{\text{th}}$  ISB. Where  $q_j^i$  shows queue length of the  $j^{\text{th}}$  GVOQ in the  $i^{\text{th}}$  ISB,  $q_j^i \geq 0$ .

In a cell slot, each ISB delivers cells to central switch fabric according to its link reservation. For example, if  $\text{LK\_RSV}^i$  is  $[2, 0, \dots, 4]$  in current cell slot, the  $i^{\text{th}}$  ISB will send 2 cells to OSB 0 and 4 cells to OSB  $(K-1)$ , but no cells are scheduled to other OSBs.

According to its queue occupancy vector, each ISB can dynamically increase/decrease its link reservation for a specific OSB by “borrowing” or “lending” links through REQ and/or REL tokens. To achieve a fast and fair link resource allocation among ISBs, we will propose a *Queue Occupancy Based Dynamic Link Reservation* (QOBDLR) algorithm in next section.

### 3.2 Link Reservation : QOBDLR Algorithm

#### Definition 1 : Link Reservation Rule.

Link reservation among  $K$  ISBs must satisfy two criteria :

- (1)  $\sum_{j=0}^{K-1} r_j^i \leq M$ , i.e. the total links reserved by the  $i^{\text{th}}$  ISB can not exceed  $M$  which is the maximum number of links at an ISB-ATMCSF interface;
- (2)  $\sum_{i=0}^{K-1} r_j^i \leq M$ , i.e. the total links reserved by all ISBs to the  $j^{\text{th}}$  OSB can not exceed  $M$  which is the maximum number of links at the ATMCSF-OSB interface.

#### Definition 2 : Link Reservation Slot, i.e. Rsv\_Slot.

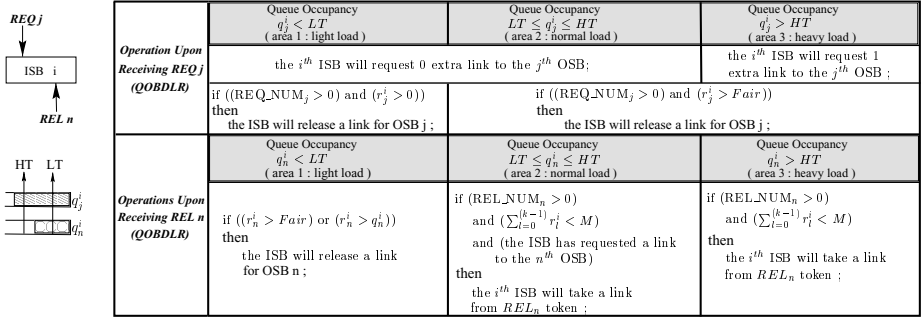
Rsv\_Slot is defined as a small time interval during which an ISB receives a pair of REQ and REL tokens. In a Rsv\_Slot, an ISB has the authority to modify its link reservation for the two OSBs which are identified by the received REQ and REL tokens. Rsv\_Slot is independent from Cell\_Slot, usually,  $\text{Rsv\_Slot} \ll \text{Cell\_Slot}$ . When a cell slot is due, every ISB delivers cells to ATMCSF according to its current link reservation vector.

QOBDLR algorithm is performed in every Rsv\_Slot. As a common model shown in Fig 2, the  $i^{\text{th}}$  ISB ( $0 \leq i < K$ ) is receiving a  $\text{REQ}_j$  and a  $\text{REL}_n$  tokens in current Rsv\_Slot, usually  $\text{REQ}_j$  and  $\text{REL}_n$  identify two different OSBs (i.e.  $j \neq n$ ). The  $i^{\text{th}}$  ISB only has the authority to modify its link reservation, i.e.  $r_j^i$  and  $r_n^i$ , for the  $j^{\text{th}}$  OSB and the  $n^{\text{th}}$  OSB in current Rsv\_Slot.

#### Operations Upon Receiving $\text{REQ}_j$ Token :

When receiving  $\text{REQ}_j$  token, the  $i^{\text{th}}$  ISB will evaluate its queue occupancy  $q_j^i$  against two thresholds : a high threshold (HT) and a low threshold (LT). Then, the  $i^{\text{th}}$  ISB decides whether to request an extra link and/or release a link to the  $j^{\text{th}}$  OSB.

If  $q_j^i > \text{HT}$ , the  $i^{\text{th}}$  ISB will request an additional link for the  $j^{\text{th}}$  OSB. Note that, if the  $i^{\text{th}}$  ISB had sent a link request before but has not obtained the desired link yet, the  $i^{\text{th}}$  ISB will not issue an extra new-born link request but just keep asking for one additional link if  $q_j^i > \text{HT}$ . And if the  $\text{REQ}_j$  token carries link requests (i.e.  $\text{REQ\_NUM}_j > 0$ ), the  $i^{\text{th}}$  ISB will schedule a single



**Fig. 2.** QOBDLR algorithm performed in a Rsv\_Slot. For example, the  $i^{th}$  ISB is receiving  $REQ_j$  and  $REL_n$  token in current Rsv\_Slot.

link release for OSB j if the ISB reserves more than  $Fair$ <sup>2</sup> links to OSB j (i.e.  $r_j^i > Fair$ ).

If  $LT \leq q_j^i \leq HT$ , the  $i^{th}$  ISB may release a link if  $REQ_j$  token carries link requests and if its link reservation for the OSB j is more than  $Fair$  links; Otherwise, the ISB will keep the same reservation as before.

If  $q_j^i < LT$ , and if  $REQ_j$  token carries link requests, the  $i^{th}$  ISB will release one of its occupied link to OSB j.

In general, a new-born link request for the  $j^{th}$  OSB will be added into  $REQ_j$  token, hence,  $REQ\_NUM_j$  will be increased by 1. On the other hand, if the  $i^{th}$  ISB releases a link to satisfy a link request in  $REQ_j$  token,  $REQ\_NUM_j$  will be reduced by 1. Usually, the link scheduled to be released can not be passed to other ISBs in current Rsv\_Slot. The  $i^{th}$  ISB will record this pending link release, and will add this released link into  $REL_j$  token when the  $i^{th}$  ISB receives  $REL_j$  token in some Rsv\_Slot(s) later.

### Operations Upon Receiving $REL_n$ Token :

When receiving  $REL_n$  token, the  $i^{th}$  ISB will evaluate its queue occupancy  $q_n^i$  with HT and LT to decide an intended modification on  $r_n^i$ .

If  $q_n^i > HT$ , the  $i^{th}$  ISB will grab an additional link for the  $n^{th}$  OSB as long as following conditions are hold : (1)  $REL_n$  token carries available links (i.e.  $REL\_NUM_n > 0$ ); (2) the total number of links reserved by the  $i^{th}$  ISB is less than M (i.e.  $\sum_{l=0}^{(k-1)} r_l^i < M$ ).

If  $LT \leq q_n^i \leq HT$ , the  $i^{th}$  ISB will take an extra link for the  $n^{th}$  OSB if following requirements are satisfied : (1)  $REL_n$  token carries available links (i.e.  $REL\_NUM_n > 0$ ); (2) the ISB has requested a link for the OSB ; (3) the total number of links reserved by the ISB is smaller than M (i.e.  $\sum_{l=0}^{(k-1)} r_l^i < M$ ).

If  $q_n^i < LT$ , the  $i^{th}$  ISB will schedule a link release if its current reservation for the  $n^{th}$  OSB is either greater than Fair (i.e.  $r_n^i > Fair$ ) or greater than its current queue occupancy (i.e.  $r_n^i > q_n^i$ ).

For the case of  $q_n^i > HT$ , it may happen that the  $i^{th}$  ISB takes a link from  $REL_n$  token but it had not sent a link request before. Under such circumstance,

<sup>2</sup>  $Fair = \lfloor \frac{M}{K} \rfloor$ , i.e. the M links at an ATMCSF-OSB interface are fairly allocated to K ISBs

we term the  $i^{th}$  ISB as a 'thieving ISB', which "steals" an available link that may have been released for a link request issued by another ISB, namely the 'victim ISB'. The 'victim ISB' who has sent a link request and is waiting for an available link may never receive its desired link if there is a 'thieving ISB' on the way snatching an available link from  $REL_n$  token. To resolve this problem, the 'thieving ISB' should send a link request for the  $n^{th}$  OSB to motivate a link release not for itself but for the 'victim ISB'. Usually, this compensated link request for the  $n^{th}$  OSB can not be inserted into  $REQ_j$  token in current Rsv\_Slot. The ISB has to record this pending link request and wait for receiving  $REQ_n$  token to send this link request to other ISBs.

For the case of  $q_n^i < LT$ , the  $i^{th}$  ISB's releasing a link for the  $n^{th}$  OSB is due to its own low traffic load and it does not need to be triggered by link requests in  $REQ_n$  token. It means that the  $i^{th}$  ISB releases a link for the  $n^{th}$  OSB without any knowledge of how many link requests are indicated in  $REQ_n$  token. In order to achieve an efficient link utilization, it is required that  $REL\_NUM_n \leq REQ\_NUM_n$ , i.e. all available links will be used up by link requests and will be needed by some ISBs. Bearing this in mind, when the  $i^{th}$  ISB releases a link for the  $n^{th}$  OSB due to  $q_n^i < LT$ , there are actually  $(REQ\_NUM_n - 1)^3$  or  $(REQ\_NUM_n - 2)^4$  link requests are expecting available links for the  $n^{th}$  OSB. Hence, the  $i^{th}$  ISB should decrease  $REQ\_NUM_n$  by either 1 or 2. However, the  $i^{th}$  ISB does not hold  $REQ_n$  token in current Rsv\_Slot. The  $i^{th}$  ISB has to record this pending reduction of link requests and wait for receiving  $REQ_n$  token to modify  $REQ\_NUM_n$ .

Due to operations for the received  $REL_n$  token, when the  $i^{th}$  ISB receives  $REQ_n$  token in some Rsv\_Slot(s) later, the ISB first has to update  $REQ\_NUM_n$  with the pending increment/decrement of link requests for the  $n^{th}$  OSB.

### 3.3 Remarks

During system initialization, all link resources are assigned to ISBs, i.e.  $\sum_{i=0}^{(k-1)} r_l^i = M$  and  $\sum_{l=0}^{(k-1)} r_l^i = M$  for  $\forall i, j$ . But, how to initialize link reservation vectors is not important because an ISB will dynamically "borrow" and/or "lend" links from/to other ISBs according to its traffic load.

Cell delivery and link reservation are independent operations. When a Cell\_Slot is due, every ISB sends cells to ATMCSF based on its current link reservation vector. But, an ISB can modify its link reservation in every Rsv\_Slot, usually  $Rsv\_Slot < Cell\_Slot$ .

In QOBDLR algorithm, the high threshold HT and the low threshold LT are predefined system parameters and are consistent after their initialization. How to make the optimal choice on the values of HT and LT is beyond this paper. In this paper, we select HT and LT as 4 and 2 for the example of an 256x256 switch. We show that the switch with QOBDLR algorithm judged by the HT and LT is able to achieve a **fair and fast** link resource allocation among ISBs.

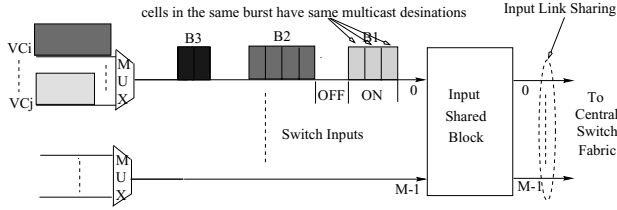
<sup>3</sup> If the  $i^{th}$  ISB has not sent a link request for the  $n^{th}$  OSB, then  $(REQ\_NUM_n - 1)$  link requests are demanding available links after the  $i^{th}$  ISB releases a link.

<sup>4</sup> If the  $i^{th}$  ISB has issued a link request for the  $n^{th}$  OSB, then  $(REQ\_NUM_n - 2)$  link requests are demanding available links after the  $i^{th}$  ISB releases a link.

## 4 Switch Performance

### 4.1 Traffic Model

Switch performance is evaluated under both uniform and non-uniform traffic. As shown in Fig 3, cells coming from different VCs are multiplexed in bursts which are interleaved to contribute as arrival traffic at every switch input. We employ the ON(active)/OFF(idle) model to describe the burst-idle process. The back-to-back cells in a ON duration belong to a same VC so that they have same destinations. No cells arrive in idle period.



**Fig. 3.** Traffic Model : Multicast Burst Traffic

To build a uniform traffic, we set a small value of MBS which is the maximum burst size (i.e. the number of cells in a ON duration). Cells' destinations are uniformly distributed among  $N$  switch outputs.

On the contrary, non-uniform traffic is featured by “hot spot” phenomenon : cells in an ISB prefer to go to some switch outputs, but not to other outputs. Three scenarios are likely to build a non-uniform burst traffic : (1) If maximum burst size MBS is very large, then cells in an ON period will keep targeting the same destinations for a relatively long time. Cell destinations are not uniformly distributed among  $N$  switch outputs in this time duration. (2) If bursts are correlated with each other, i.e. cells in a successive ON periods have the same destinations. Even though MBS may be small, cells accumulated in several bursts will make the traffic non-uniformly distributed among  $N$  output ports. (3) In an extreme case, an ISB has cells only destined to a specific OSB, but has no cells target for other ISBs. This is so called “1 ISB  $\rightarrow$  1 OSB hot spot” traffic.

In this paper, VBR source is used to generate the ON-OFF traffic with following traffic parameters : MBS, i.e. the maximum burst size; PCR, i.e. the peak cell rate (the number of cells/sec) which satisfies that  $PCR \leq LCR$ ; LCR, i.e. the line cell rate which approximates  $\frac{15520000}{53 \times 8} = 366,793 (cells/sec)$  for an OC-3 link; ACR, i.e. the average cell rate. We define  $F_{out}$  as the fanout of a cell.  $F_{out}$  has a uniform distribution from 0 to  $C_{max}$ . The average fanout load  $F = (C_{max} + 1)/2$ , where  $C_{max}$  is the maximum copies allowed for a multicast cell. The effective input load is defined as  $\rho = ACR * F / LCR$ ,  $\rho \leq 1$ .

### 4.2 Performance Evaluation

Using OPNET/MIL3 simulation platform [11], we simulated an 256x256 ( $N = 256$ ) switch which consists of 8 ISBs and 8 OSBs ( $K = 8$ ). Each ISB/OSB is of

size  $32 \times 32$  ( $m = M = 32$ ). Meantime, we compare the proposed switch with a switch design in our previous work [9]<sup>5</sup>. As a comparison, we also simulated an  $256 \times 256$  output queued switch under the same traffic scenarios. There are two reasons for us to select OQ switch as a comparison reference : (1) OQ switches proved to maximize throughput and optimize latency under any traffic pattern; (2) in the literature so far, few of existing switch designs is dedicated for a distributed large scale switch and is investigated under any traffic condition. Therefore, we believe that it is fair and effective to compare our switch design with an OQ switch under same traffic patterns.

We investigate switch performance under both uniform and non-uniform traffic. Following performance statistics are estimated :

- **Throughput** : switch throughput which is statistically measured on N switch outputs ;
- $D_{E-to-E}$  : the average end-to-end cell delay ( # of cell slots) which is defined as the latency for a cell going through the switch;
- $D_{ISB}$  : the average cell delay in ISBs ( # of cell slots); ATMCSF is assumed to deliver cells from input shared links to output shared links in a cell slot, hence, end-to-end cell delay  $D_{E-to-E}$  is resulted from two parts : the *cell delay in ISBs (i.e.  $D_{ISB}$ )*, and the *cell delay in OSBs*.
- $S_{OSB}$  : the average occupancy of an OSB (# of cells); an OSB has  $M$  output queues,  $S_{OSB}$  indicates the total number of cells waiting in an OSB.

Table 1 shows switch performance under **uniform traffic** with different input load  $\rho$ . Both unicast traffic and multicast traffic are applied. In unicast uniform traffic, every cell arriving at a switch input only carries one destination. But, in multicast uniform traffic, a coming cell may have multiple destinations. Cells' destinations are uniformly distributed among N switch outputs.

In the simulation, the proposed enhanced switch performs link reservation with the slowest rate, i.e.  $Rsv\_Slot = Cell\_Slot$ . Hence, in a cell slot, a token can only pass through one ISB. Any faster link reservation rate (i.e.  $Rsv\_Slot < Cell\_Slot$ ) will result in better performance than what we simulated here. The values of HT and LT are selected as 4 and 2 respectively.

Under uniform traffic, both the Switch in [9] and the proposed enhanced switch can achieve a comparable performance as the OQ switch. On **throughput** performance, the OQ switch always obtains the maximized throughput  $\rho$ , while our switch designs closely approach to the OQ switch with less than 0.6% throughput degradation. In general, the **end-to-end cell delay**  $D_{E-to-E}$  increases with input load  $\rho$ . Compared with the lower bound of  $D_{E-to-E}$  achieved in the OQ switch, the  $D_{E-to-E}$  in our switch designs cause 2~20 more cell slots. Longer cell delay is due to lower throughput. Since employing dynamic link reservation, the proposed switch outperforms the Switch in [9] in the performance of throughput, cell delay and delay jitter. In addition, Table 1 shows that the end-to-end cell delay in our designs is mainly due to the latency in the OSBs

<sup>5</sup> The switch in [9] has a similar switch architecture as the proposed enhanced switch, but ISBs are isolated from each other without any connection. In a cell slot, an ISB is only responsible for delivering cells to a certain OSB according to a round robin one-to-one mapping from K ISBs to K OSBs. The switch had been demonstrated to be able to obtain a good performance under uniform traffic, but it has a weakness to support non-uniform traffic.

rather than the cell delay in ISBs. It is a good feature of the proposed switches because cells are forwarded to OSBs in a faster manner and most of the cells are queued in OSBs. Hence, the proposed switches explicit a capability to mimic the OQ switch, and OSBs may incorporate per VC queueing with appropriate cell schedulers to provide QoS guarantees. This is a subject of our ongoing work.

		Uniform, Unicast, Burst Traffic ( F = 1, Fout = 1, MBS = 16 )			Uniform, Multicast, Burst Traffic ( F = 4 , Cmax = 16, MBS = 16 )		
		<i>OQ Switch</i>	<i>Switch in [9]</i>	<i>The proposed switch</i> Rsv_Slot = Cell Slot ; HT = 4 , LT = 2	<i>OQ Switch</i>	<i>Switch in [9]</i>	<i>The proposed switch</i> Rsv_Slot = Cell Slot ; HT = 4 , LT = 2
$\rho = 0.99$	Throughput	0.99	0.98440	0.98495	0.99	0.98903	0.98930
	$D_{E-to-E}$ ( Min, Max )	47.25 ( 1.3, 197 )	65.52 ( 6.95, 242 )	62.3 ( 1.7, 222.8 )	25 ( 1.0, 156 )	35 ( 5.6, 203 )	34 ( 2.0, 189.2 )
	$D_{ISB}$ ( Min, Max )	N/A	22.8 ( 1.0, 91 )	20.2 ( 1.0, 63 )	N/A	4.9 ( 1.0, 18.1 )	4.2 ( 1.0, 25.7 )
	$S_{OSB}$ ( Min, Max )	N/A	1362 ( 1152, 1545 )	1357 ( 1120, 1565 )	N/A	768 ( 591, 969 )	758 ( 572, 964 )
$\rho = 0.90$	Throughput	0.90	0.89866	0.89892	0.90	0.89913	0.89932
	$D_{E-to-E}$ ( Min, Max )	11.38 ( 1.1, 104 )	18.90 ( 6.25, 116 )	17.93 ( 2.0, 119 )	9.5 ( 1.0, 97 )	15.3 ( 3.5, 133 )	14.2 ( 1.5, 117 )
	$D_{ISB}$ ( Min, Max )	N/A	5.8 ( 1.0, 30 )	5.0 ( 0.1, 20 )	N/A	5.9 ( 1.0, 17 )	4.7 ( 1.0, 26 )
	$S_{OSB}$ ( Min, Max )	N/A	331.5 ( 249, 423 )	316.5 ( 213, 409 )	N/A	274 ( 201, 355 )	263.5 ( 172, 365 )
$\rho = 0.70$	Throughput	0.70	0.69918	0.69924	0.70	0.69949	0.69988
	$D_{E-to-E}$ ( Min, Max )	4.01 ( 1.0, 46.5 )	8.4 ( 6.6, 70 )	5.2 ( 2.0, 67 )	3.6 ( 1.0, 38 )	8.1 ( 2.0, 76 )	4.6 ( 2.0, 63 )
	$D_{ISB}$ ( Min, Max )	N/A	4.5 ( 1.0, 14 )	2.2 ( 1.0, 10 )	N/A	4.5 ( 1.0, 12.8 )	1.3 ( 1.0, 12.1 )
	$S_{OSB}$ ( Min, Max )	N/A	100 ( 71, 133 )	87 ( 59, 116 )	N/A	86.5 ( 60, 117 )	75 ( 52, 102 )
$\rho = 0.50$	Throughput	0.50	0.49953	0.49987	0.50	0.49963	0.49994
	$D_{E-to-E}$ ( Min, Max )	2.25 ( 1.0, 26.7 )	3.1 ( 2.0, 48.2 )	2.9 ( 2.0, 38.2 )	2.0 ( 1.0, 23.5 )	3.6 ( 1.0, 47 )	2.3 ( 1.0, 33 )
	$D_{ISB}$ ( Min, Max )	N/A	2.1 ( 1.7, 13 )	1.2 ( 1.7, 12 )	N/A	2.1 ( 1.0, 10.95 )	1.05 ( 1.0, 11.2 )
	$S_{OSB}$ ( Min, Max )	N/A	45 ( 30, 64 )	35 ( 23, 52 )	N/A	38.3 ( 22.3, 56.7 )	29.6 ( 18.8, 46.8 )

**Table 1.** Performance Comparison under Uniform Traffic with Different Input Load  $\rho$ . Both unicast and multicast traffic are applied.

Now, we investigate switch performance under non-uniform traffic in Table 2. The traffic applied is "1 ISB  $\rightarrow$  1 OSB HotSpot Traffic" : the input traffic to the  $i^{th}$  ISB is dedicated to the  $i^{th}$  OSB, i.e. the cells in the  $i^{th}$  ISB only target the  $i^{th}$  OSB but no cell goes to other OSB. Both unicast traffic pattern and multicast traffic pattern are introduced.

Table 2 shows that the proposed enhanced switch can achieve a comparable performance as the OQ switch, but the Switch in [9] suffers a lot and can not survive under this non-uniform traffic. The reason for that is, the Switch in [9] only allows an ISB to deliver cells to its matched OSB according to a round robin one-to-one mapping. If the ISB does not have cells to go to the OSB, other

		<b>1 ISB - 1 OSB HotSpot, Unicast, Burst Traffic</b> ( F = Fout = 1 , Cmax = 16, MBS = 40, HotSpotBstLen = 20*5 )			<b>1 ISB - 1 OSB HotSpot, Multicast, Burst Traffic</b> ( F = 4 , Cmax = 16, MBS = 40, HotSpotBstLen = 20*5 )		
		<i>OQ Switch</i>	<i>Switch in [9]</i>	<i>The proposed switch</i> Rsv_Slot = Cell Slot ; HT = 4 , LT = 2	<i>OQ Switch</i>	<i>Switch in [9]</i>	<i>The proposed switch</i> Rsv_Slot = Cell Slot ; HT = 4 , LT = 2
$\rho = 0.99$	Throughput	0.99	0.13234	0.98593	0.99	0.53693	0.98986
	$D_{E-to-E}$ (Min, Max)	152 ( 1.0 , 315 )	1431 ( 39 , 2087 )	217 ( 2.0 , 397 )	113 ( 1.0 , 338 )	337 ( 13.5 , 624 )	157 ( 2.0 , 482 )
	$D_{ISB}$ (Min, Max)	N/A	1370 ( 1.0 , 1964 )	88 ( 1.0 , 178 )	N/A	313 ( 1.0 , 596 )	1.1 ( 1.0 , 11.0 )
	$S_{OSB}$ (Min, Max)	N/A	84 ( 5.0 , 113 )	4123 ( 3771 , 4459 )	N/A	188 ( 61.0 , 169 )	3842 ( 2576 , 5028 )
$\rho = 0.90$	Throughput	0.90	0.13098	0.89799	0.90	0.53381	0.89990
	$D_{E-to-E}$ (Min, Max)	72 ( 1.0 , 178 )	1070 ( 31 , 1710 )	125 ( 2.0 , 227 )	73 ( 1.0 , 188 )	289 ( 13.5 , 429 )	112 ( 2.0 , 246 )
	$D_{ISB}$ (Min, Max)	N/A	986 ( 7.0 , 1667 )	43 ( 1.0 , 147 )	N/A	267 ( 1.0 , 375 )	1.1 ( 1.0 , 10 )
	$S_{OSB}$ (Min, Max)	N/A	62 ( 5.5 , 87 )	2580 ( 2160 , 2980 )	N/A	112 ( 64 , 178 )	3175 ( 2048 , 4738 )
$\rho = 0.70$	Throughput	0.70	0.12984	0.69988	0.70	0.54085	0.69992
	$D_{E-to-E}$ (Min, Max)	26 ( 1.0 , 75 )	853 ( 30 , 1457 )	47 ( 2.0 , 97 )	15.4 ( 1.0 , 38.2 )	97 ( 11.7 , 189 )	26.3 ( 2.0 , 57.3 )
	$D_{ISB}$ (Min, Max)	N/A	837 ( 5.0 , 1412 )	19 ( 1.0 , 65 )	N/A	88 ( 1.1 , 133 )	1.1 ( 1.0 , 7.0 )
	$S_{OSB}$ (Min, Max)	N/A	12.0 ( 5.5 , 21 )	597 ( 436 , 770 )	N/A	110 ( 68 , 203 )	786 ( 130 , 1289 )
$\rho = 0.50$	Throughput	0.50	0.12746	0.49993	0.50	0.49925	0.49995
	$D_{E-to-E}$ (Min, Max)	11 ( 1.0 , 23 )	553 ( 2.0 , 718 )	17.8 ( 2.0 , 37 )	6.0 ( 1.0 , 13.4 )	13.2 ( 10.0 , 22 )	10.7 ( 1.0 , 21.3 )
	$D_{ISB}$ (Min, Max)	N/A	539 ( 4.0 , 694 )	8.7 ( 1.0 , 17 )	N/A	8.2 ( 2.0 , 186 )	1.0 ( 1.0 , 6.0 )
	$S_{OSB}$ (Min, Max)	N/A	8.0 ( 6.5 , 9.3 )	277 ( 120 , 573 )	N/A	106 ( 45 , 178 )	287 ( 21.3 , 379 )

**Table 2.** Performance Comparison under Non-uniform Traffic with Different Input Load  $\rho$ . We apply “1 ISB - 1 OSB HotSpot” traffic with two patterns — unicast traffic, and multicast traffic.

ISBs have no authority to send cells to the starved OSB. Under “1 ISB  $\rightarrow$  1 OSB HotSpot Traffic”, an ISB only has cells to be delivered in 1 out of every K cell slots. Hence, performance of the switch in [9] is very poor. It is observed that, when input load  $\rho = 0.99$ , the switch in [9] gains no more than 14% throughput under unicast traffic and obtains 54% throughput under multicast traffic. GVOQ scheme is effective in the multicast traffic so that it results in a higher throughput than unicast traffic. Since more and more cells are backlogged in ISBs, the Switch in [9] suffers an increasing cell delay.

The proposed enhanced switch, unlike the Switch in [9], utilizes dual round robin dynamic link reservation which can adapt the traffic loading to perform an efficient link resources allocation among ISBs. Starvation of OSBs is relaxed. It shows that the proposed switch can approach to a very similar performance as the OQ switch on throughput, end-to-end cell delay  $D_{E-to-E}$ , and delay jitter (Min, Max) of  $D_{E-to-E}$ . In the proposed switch, most cells are queued in OSBs,

and  $D_{E-to-E}$  is mainly resulted from the cell delay in OSBs. As we mentioned, it is a good feature of the proposed switch because OSBs may incorporate per VC queueing with appropriate cell schedulers to provide QoS guarantees. In summary, the proposed switch demonstrates a promising capability to achieve a high performance like the OQ switch under both uniform and non-uniform traffic scenarios. Compared with the OQ switch, the proposed enhanced switch eliminates  $N$  times speedup which, however, is necessary in the OQ switch.

## 5 Conclusion

In this paper, we present a scalable multicast packet switch with a modular switch architecture and a distributed dynamic link reservation algorithm. The switch benefits from input and output link sharing. It resolves output contention and virtually eliminates speedup requirement for the central switch fabric. QOBDLR is a distributed algorithm in which an ISB can dynamically increase/decrease its link reservation for a specific OSB according to its local available information. Arbitration complexity is only  $O(1)$ . QOBDLR can achieve a fast and fair link resource allocation among ISBs. Simulations on an  $256 \times 256$  switch demonstrate that the proposed switch can achieve a comparable performance as the output queued switch under any traffic pattern.

Apart from the preliminary results discussed in this paper, theoretical work on the choice of HT and LT for QOBDLR algorithm is our ongoing work.

## References

1. M. H. Guo, R. S. Chang, *Multicast ATM Switches : Survey and Performance Evaluation*, Computer Communication Review, Vol 28, No. 2, April 1998, pp. 98-131.
2. J. Turner, N. Yamanaka, *Architecture Choices in Large Scale ATM Switches*, WUCS 97-21, May 1997.
3. H. J. Chao, B. S. Choe, *Design and Analysis of A Large-Scale Multicast Output Buffered ATM Switch* IEEE/ACM Trans. on Networking, Vol. 3, No. 2, April 1995, pp. 126-138.
4. H. J. Chao, B. S. Choe, J. S. Park, N. Uzun, *Design and Implementation of Abacus Switch : A Scalable Multicast ATM Switch*, IEEE J. on Select. Areas in Commun., Vol. 15, No. 5, June 1997, pp. 830-843.
5. N. McKeown, V. Anantharam, J. Walrand, *Achieving 100% Throughput in an Input-Queued Switch*, Proc. of IEEE Infocom96, March 1996.
6. A. Mekikittikul, N. McKeown, *A Practical Scheduling Algorithm to Achieve 100% Throughput in Input-Queued Switches*, Proc. of IEEE Infocom98, April 1998.
7. S.-T. Chuang, A. Goel, N. McKeown, B. Prabhakar, *Matching Output Queueing with Combined Input and Output Queueing*, Proc. of IEEE Infocom99, March, 1999.
8. B. Prabhakar, N. McKeown, *Designing A Multicast Switch Scheduler*, Proc. of the 33<sup>th</sup> Annual Allerton Conference on Communication, Control and Computing, October 1995.
9. F. Chen, N. Uzun, A.N. Akansu, *A Large Scale Multicast ATM Switch With Input and Output Link Sharing*, Proc. of IEEE Globecom'99, Rio de Janeiro, Brazil, December 1999, pp. 1251-1255.
10. F. Chen, N. Uzun, A.N. Akansu, *A Scalable Multicast ATM Switch using Link Sharing and Prioritized Link Reservation*, Proc. of IEEE ICCCN'99, Boston, Massachusetts, October 1999, pp. 218-222.
11. OPNET by MIL3 Inc., Washington, DC 20008.



# An Interworking Call Control Solution for a Multidiscipline Switch

Pertti Raatikainen<sup>1</sup> and Sami Raatikainen<sup>2</sup>

<sup>1</sup> VTT Information Technology, Telecommunications, P.O. Box 1202, FIN-02044 VTT,  
Finland

`pertti.raatikainen@vtt.fi`

<sup>2</sup> Helsinki University of Technology, Telecommunications Software and Multimedia  
Laboratory, P.O. Box 5400, FIN-02015 HUT, Finland

`sami.raatikainen@acta.fi`

**Abstract.** The development of multimedia services is pushing networking technologies towards broadband solutions. A lot of research and development effort has been spent on finding a universal transport solution to carrying the different services to end-users. However, it looks that there will be no unifying transport solution to overcome the burden of networking diversity. This implies that the future services are delivered to users over heterogeneous networks. Therefore, interworking between the different networks and between the different signalling protocols becomes an essential part of the game. This paper introduces an interworking signalling solution for a switching equipment that integrates PDH time-slot and ATM cell switching. The switching platform is based on a multidiscipline switching fabric previously utilized in ATM switching applications.

## 1 Introduction

The rapid growth of Internet and advent of interactive multimedia communications is pushing the development of transport technologies towards heterogeneous networking. Services that traditionally have been carried over circuit or packet switched networks are seen to cross boundaries of the different network. This means that the future networks must be able to support services that have not originally been designed for them and, additionally, they must be able to communicate with other networks deploying different transport solutions. This places a major challenge for the equipment manufacturers and service providers. Manufacturers have to develop equipment having capabilities to interwork seamlessly with equipment supporting other networking technologies. Service providers have to satisfy a wide range of customer needs and try to guess the customers' future needs as well. So, a service provider needs an easy-to-extend environment.

End-users should be able to communicate with each other regardless of the underlying transport network. This means that the end-user needs universal access, i.e., access to other users as well as to all kinds of services. Therefore, network

elements, such as switches, have to implement capabilities to communicate with different networks and support a diverse set of call and connection control schemes. The key issue, until all manufactures have the same technology if ever, is interworking.

Interworking between different networks and between different signalling protocols implies mappings and information conversions between the communicating parties. Both the hardware platform and the software framework have to implement different signalling protocol stacks with different signalling procedures and be able to interconnect them. In addition, different kinds of resource allocations and verifications, e.g., VPI/VCI and time-slots, have to be provided. Thus, a flexible framework is needed to enable all this in one software package.

In this paper, we introduce an interworking signalling solution for a switching equipment that integrates ATM (Asynchronous Transfer Mode) cell and PDH (Plesiochronous Digital Hierarchy) time-slot switching. The interworking is performed between narrowband and broadband signalling protocols, i.e., DSS1, ISUP, DSS2 and BISUP. The solution is based on an IN (Intelligent Networks) solution [10] that was considered an ideal basis for the work. The physical switching platform is based on a multidiscipline switch that integrates circuit, cell and packet switching into a single fabric [11]. The signalling architecture is based on the solution introduced in [12]. Chapter 2 introduces the IN concept and Chapter 3 presents the developed interworking software solution. Chapter 4 discusses the future work and Chapter 5 has the concluding remarks.

## 2 IN Concept and Interworking

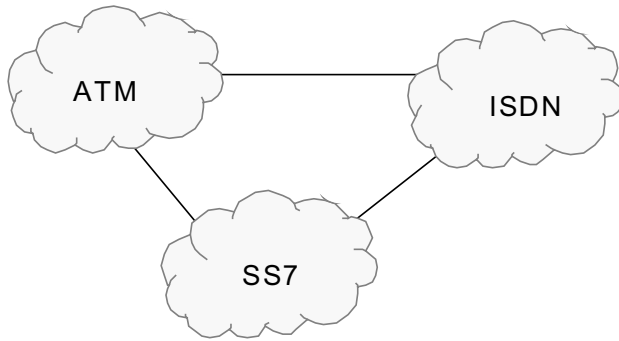
Signalling protocols have evolved quite slowly along with terminals and network nodes. Development of ISDN networks introduced separate protocols for user and network signalling, because network nodes were assumed to be more intelligent than user terminals. The separation of signalling protocols based on different interfaces led to the need for mapping functions to transfer information between the involved protocols (e.g. DSS1 and ISUP) [13]. This can be seen as one form of interworking.

Interworking between the different networking technologies is a much more complex problem. In order to interconnect narrowband and broadband networks, more intelligent devices equipped with interworking capabilities are required. Even the networks implementing different protocol versions of the same signalling standard (e.g. ITU-T ISUP and ANSI ISUP) require interworking capabilities. Additionally, interworking between different interfaces within a certain type of network has to be supported.

Intelligent Networks (IN) [5] concept provides an open service platform where services can be located to selected network nodes and still are accessible from all corners of the network. The IN concept forms a multiservice environment enabling support for a large variety of connections with easily extensible IN services. The IN standards considered quite early interworking and suggested a general call model [5].

This environment can be utilized in realizing interworking between different sorts of networks.

The IN capabilities have successfully been deployed in today's digital telecommunications networks, such as ISDN (Integrated Services Digital Network), that utilize ITU-T's SS7 signaling (often implemented as a separate signalling network). Broadband technologies, such as ATM, can provide sophisticated control mechanisms and adequate transport capacity for a large set of services. Today, these technologies are mainly implemented in SDH (Synchronous Digital Hierarchy) based backbone networks, but with new and advanced access technologies, such as xDSL (Digital Subscriber Line), broadband services will become available to end-users who have a twisted pair connection to the trunk network. Figure 1 shows possible interconnections between different networks.

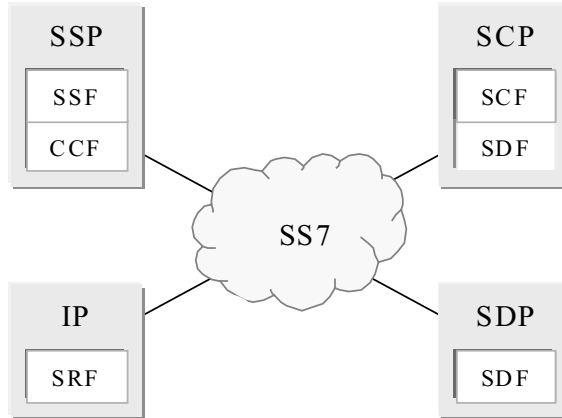


**Fig. 1.** Interconnections between different network technologies.

The IN concept defines a network service control architecture for open, distributed and service-independent communications. The architecture allows different services to be offered to customers independent of the underlying network technology. It also defines a service-oriented functional architecture that enables the provision of a set of generic service components of which new telecommunications services can be constructed. From the service point of view, the IN network can be considered as an overlay network, i.e., service control is separated from service switching functions.

The two basic components of the IN functional entities are Service Switching Point (SSP) and Service Control Point (SCP). SSP provides users with network access, performs switching functionality and allows users to access IN services. It detects IN service requests (e.g. a prefix of a number) and communicates with other physical IN entities. SSP contains Call Control Functions (CCF) and Service Switching Functions (SSF). CCF provides call/service processing and control. SSF provides a set of functions required for interaction between CCF and SCF (Service Control Functions). SCF is located in SCP that contains the service logic programs and data that are used in providing IN services.

Examples of other functional entities are Intelligent Peripheral (IP) and Service Data Point (SDP). IP provides special resources with Specialized Resource Function (SRF). SDP contains Service Data Function (SDF) for real-time access to customer and network data. Figure 2 illustrates some of the IN entities.



**Fig. 2.** Some physical entities of the IN architecture.

### 3 Interworking Call Control Solution

The ambitious goal of this work was to develop a modular software architecture, which ultimately enables an "any-to-any" interworking capability with any manufacturer's switch. The software can act as a SSP equipped with CCF and SSF functionality (see Fig. 2). It supports user signalling and bearer connection control at ISDN (Q.931) and at ATM (Q.2931, UNI3.1, UNI4.0) interfaces as well as network signalling at SS7 (ISUP) and ATM (B-ISUP) interfaces.

#### 3.1 Main Software Modules

The interworking signalling implementation consists of software modules, compiled separately and linked together to form the entire software package. The major modules are the ICC (Interworking Call Control) module and the signalling stack modules (see Fig. 3). The ICC module includes the basic and generic CC (Call Control) and SCC (Switching Call Control) modules. The CC module implements a service switching point (SSP) which offers different call control functions (CCF), basically call models, for the originating (CC-O) and terminating (CC-T) sites of a call. The SCC module interconnects these call models and provides interworking functions and protocol specific bearer connection control functions. The FCF (Fabric Control Functions) module includes all the functions needed to control physical hardware connections. FCF utilizes API (Application Programming Interface) of the physical switch.

The signalling stacks (DSS1, DSS2, ISUP and BISUP) function independently below the interworking call control layer. Each stack implements signalling and necessary lower layer protocols. The signalling stacks communicate with the call control application in the CC module via a common signalling interface, named SIGIF. Since the different signalling protocols use the same primitives, it has been possible to develop this common and well-defined "non-fat interface", primarily based on DSS1.

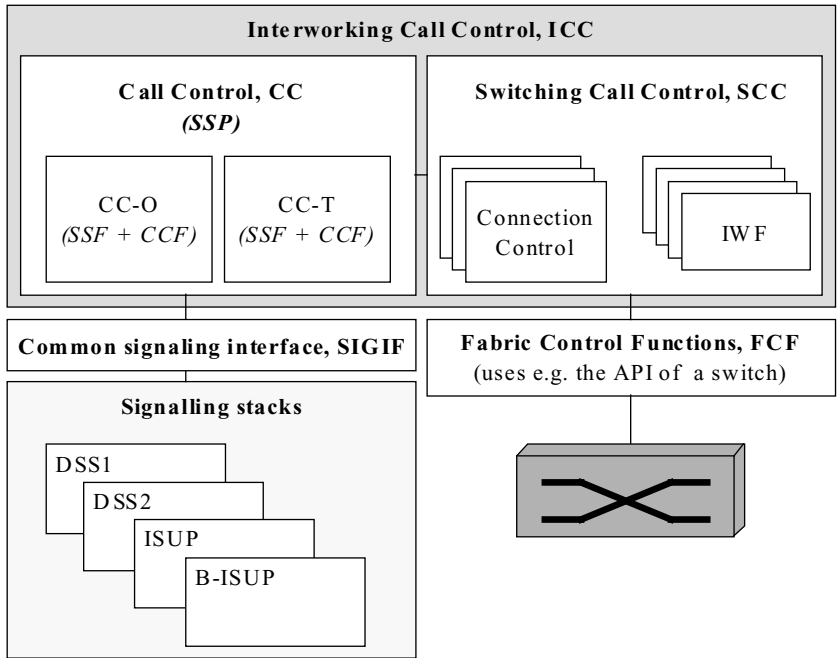


Fig. 3. Software modules.

3.2 Interworking Call Control

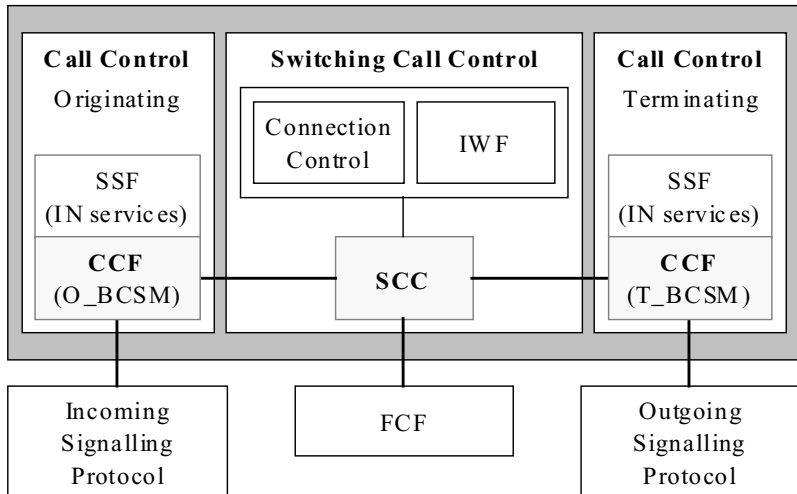
The interworking call control block, illustrated in Fig. 4, is composed of separate and independently interconnected modules for each subsystem (ATM, PDH and ISDN). Although these modules can be seen to function independently, they rely on other modules' support (e.g. SCC relies on the knowledge that a message sequence is checked by the signalling protocols).

The Call Control (CC) module, which is common to all signalling protocols and network types, can be considered as a central component. It is divided into two submodules CC-O and CC-T. CC-O describes the call control model for all incoming calls and CC-T the corresponding model for all outgoing calls. The implementation of these models is based on the definitions given in ITU-T's IN specifications [5]. SCC

interconnects these two CC modules with the connection control and interworking functions. This implementation structure allows separation of call and connection control making the software more scaleable.

When a new call is requested, the required call control and interworking functions are activated. A “CC-O  $\diamond$  SCC  $\diamond$  CC-T” connection is formed for each established call. Since CC-O and CC-T are common to all protocols, they retain the same structure for all calls. SCC, on the contrary, is created for each call individually and it describes all protocol and network dependent functions related to the call.

CC-O and CC-T process control information transparently which makes the CC module a generic one. This transparency is the primary reason for the need of the SIGIF interface. The CC module processes SIGIF primitives without deeper knowledge of their content; it knows only the primitive types and possibly some other information. The SCC module, on the other hand, knows all details of the SIGIF primitives and is capable of forming new ones with new parameters.



**Fig. 4.** Interworking call control.

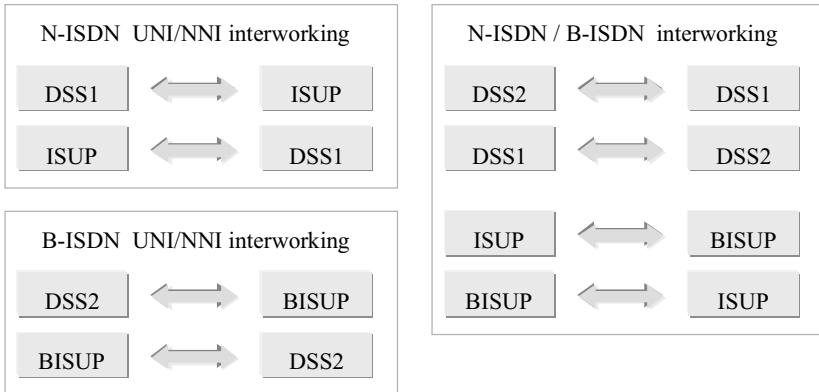
The connection control functions in SCC know which methods to invoke in FCF to have the requested interconnection (e.g. to interconnect an ISDN channel with an ATM virtual channel). FCF “hides” the implementation of the physical switching fabric and hence a new fabric can be implemented just by adding a new FCF. By the use of this approach two FCF-solutions have been developed, one that uses GSMP (General Switch Management Protocol) [9] to manage a pure ATM switch and another that uses API of the deployed multidiscipline switch [11] to manage multiple switching schemes.

Since the interworking procedures depend on the originating site of the call and either end can function as the originator, different implementations are required for the

two cases (see Fig. 5). All applied signalling protocols are based on ITU-T standards and thus the developed interworking solution is also based on ITU-T standards. Some modifications suggested by ETSI have been considered.

Interworking between the following signalling protocols (illustrated in Fig. 5) are supported:

1. Broadband interworking:
  - DSS2 - BISUP [1, 6, 8]
2. Narrowband interworking:
  - DSS1 - ISUP [4]
3. N-ISDN - B-ISDN interworking
  - DSS1 - DSS2 [8]
  - ISUP - BISUP [2, 7]



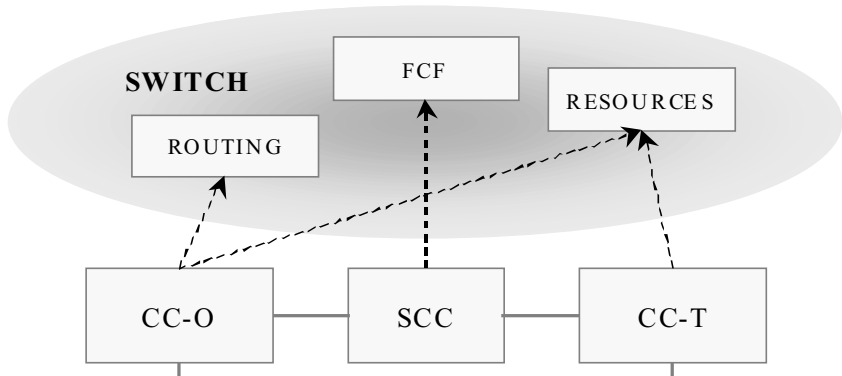
**Fig. 5.** Supported interworking procedures.

### 3.3 Resource Allocation and Routing

The starting point in the resource allocation was that a multidiscipline switch has to support different kinds of resources, e.g., ATM, conventional PDH and ISDN resources. All these with different traffic parameters have to be controlled which poses the problems of separating the resources and related procedures. In the case of interworking, different resources are tied to a connection and these should implement characteristics to support interoperability. Three resource entities (i.e. ATM, PDH and ISDN port type) were developed to describe port specific resources. They are used for controlling, e.g., ATM virtual channels, PDH time-slots and resources of the switching fabric. Each entity describes its own set of traffic characteristics and includes parameter lists and connection identifier tables, such as VPI/CVI tables.

Routing refers here to routing of control messages and it may be static (i.e. routing tables configured by an operator) or dynamic (i.e. routing tables configured by a routing protocol). Routing has been implemented as a separate object, equipped with a unique interface and related functionality. This compact interface is relatively easy to include into an already existing routing solution. Thus, it is possible to implement a number of different routing solutions.

The SWITCH component is designed with a facade design pattern [3] and it provides a unified interface to all system resources. SWITCH controls all the system resources and routing operations as well as access to them (see Fig. 6). It is implemented using a singleton design pattern [3], which ensures that there exists only one instance of it and the software provides a global point of access to it.



**Fig. 6.** Accessing services through the SWITCH component.

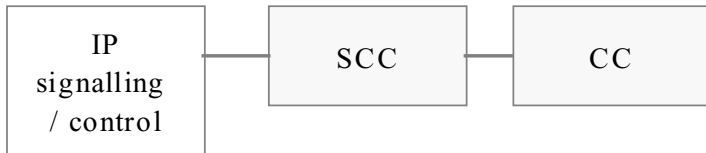
SWITCH initializes all software (including signalling stacks), controls numerous individual software components and directs incoming requests to relevant resource or routing entities. For example, a request coming from the CC module is directed to the relevant (i.e. ATM, PDH or ISDN) port entity, which reserves the needed resources (e.g. VPI/VCI). The resource control operations are hidden from the CC module, because only the port entities process the parameters of the connections. Thus call control can function independent of the physical switching fabric. When establishing or releasing connections, the SWITCH component provides the address of FCF and by using this address SCC can access functions needed to control the connections. The SWITCH component handles also routing requests and thus no other component can access routing resources.

## 4 Future Work

The deployed switching platform supports integration of ATM, PDH and IP communications and to have real convergence between these networks, interworking



with IP networks have to be considered. Interworking with IP networks requires functionality to establish and control connections between IP and ATM or IP and PDH networks. This implies that software to implement “IP signalling” is needed. The developed modular architecture allows at least one way of adding IP signalling and connection control to the developed software. Fig. 7 illustrates integration of IP control to the developed software solution.



**Fig. 7.** Integrating IP control to the control software.

IP signalling and control module must include functions necessary for setting up and clearing connections also in the other networks involved with interworking, (i.e. ATM SVCs or PDH 64 kbps circuits). This module communicates with the SCC module, which will be able to convert and structure the information needed by the signalling protocol of the other communicating network. In the first phase, a connection would be needed from the IP switch control block to SCC. Then SCC could include functionality to make connections to the other network. Additionally, IP routing should also be implemented as a separate module.

## 5 Conclusions

Diversity in existing networking solutions and the development of multimedia services is leading to heterogeneity in networking. Since services are transported over different kinds of networks, it becomes essential to implement interworking facilities into the network. This article has introduced an interworking signaling solution for a switching equipment that integrates ATM cell and PDH time-slot switching.

The switching fabric is based on a switching solution, called SCOMS, that integrates ATM, conventional PDH and in the near future also IP switching into a single fabric. The key issue is to have reliable operation between the different systems and interchange signalling information. The developed system implements each subsystem separately, but the call control model is common to all integrated switching schemes.

The developed interworking solution utilises the general call models introduced in ITU-T's Intelligent Network (IN) concept. These models, defining separate call models for the originating and terminating site of a call, were adopted to our solution and every connection uses them in the same way. Modules implementing these call models function together as a Service Switching Point (SSP) and they are linked

together with a Switching Call Control (SCC) module to enable interworking. A separate Fabric Control Function (FCF) was implemented to interpret control messages to the physical switch. The FCF module hides the physical switch implementation enabling a new fabric to be installed just by replacing the existing FCF by a new one.

The future work includes implementation of IP switching into the deployed multidiscipline switch platform. Separate software modules will be needed for routing and interworking solutions. Interworking requires implementation of "IP signalling" which must include capabilities to set up and clear connections also in the other networks that are involved with interworking.

## References

1. ETSI ETS 300 495: Interworking between B-ISUP and DSS2.
2. ETSI ETS 300 496: Interworking between B-ISUP and ISUP.
3. Gamma, E., Helm, R., Johnson, R., Vlissides, J. Design Patterns: Elements of Reusable Object-Oriented Software. Addison-Wesley Publishing Company. Massachusetts (1995). ISBN 0-201-63361-2.
4. ITU-T Recommendation Q.699 (09/97): Interworking between ISDN access and non-ISDN access over ISDN User Part of Signalling System No. 7.
5. ITU-T Recommendation Q.1204 (03/93) - Intelligent network distributed functional plane architecture.
6. ITU-T Recommendation Q.2650 (02/95): Broadband-ISDN, interworking between Signalling System No. 7 - Broadband ISDN user part (B-ISUP) and Digital Subscriber Signalling System No. 2 (DSS 2).
7. ITU-T Recommendation Q.2660 (02/95): Broadband integrated services digital network (B-ISDN) - Interworking between Signalling System No. 7 - Broadband ISDN user part (B-ISUP) and Narrow-band ISDN user part (N-ISUP).
8. ITU-T Recommendation Q.2931 (02/95): Broadband Integrated Services Digital Network (B-ISDN) - Digital Subscriber Signalling System No. 2 (DSS 2) - User-Network Interface (UNI) - Layer 3 specification for basic call/connection control.
9. Newman P., Edwards W., Hinden R., Hoffman E., Liaw F. C., Lyon T., Minshall G.: Ipsilon's General Management Protocol - Version 1.1. IETF RFC 1987, Aug. 1996.
10. Puro V-M., Koponen P., Räsänen J., Nummisalo P., Martikainen O.: TOVE in Universal Mobile Telecommunications Systems. Proceedings of the 2nd Workshop on Personal Wireless Communications. IFIP TC6. Ed. By O. Spaniol, J. Slavik and O. Drobniak, Frankfurt am Main (Germany), Dec. 1996, pp. 103 - 111.
11. Raatikainen K. P. T.: Multidiscipline Switching for Delivering Multimedia Services. Proceedings of ICT'99. Cheju (Korea), June 1999, Vol. 2, pp. 86 -90.
12. Raatikainen K.P.T.: Interworking Support in a Multidiscipline Switch. Proceedings of Broadband Communications Conference 1999 (BC'99), Convergence of Network Technologies. Hong Kong, Nov. 1999, pp. 395 - 404.
13. Venieris I., Hussman H.: Intelligent Broadband Networks. John Wiley & Sons (1998). ISBN 0-471-98094-3.

# Generic Multicast Transport Services: Router Support for Multicast Applications

Brad Cain<sup>1</sup> and Don Towsley<sup>2</sup>

<sup>1</sup> Network Research, Nortel Networks  
Billerica, MA 01821, USA  
bcain@nortelnetworks.com

<sup>2</sup> Department of Computer Science, University of Massachusetts,  
Amherst MA 00103, USA  
towsley@cs.umass.edu

**Abstract.** The development of scalable end-to-end multicast protocols is a tremendous challenge because of the problems in feedback implosion and transmission isolation. In this paper we describe a set of simple services, called *Generic Multicast Transport Services (GMTS)*, which are implemented in routers for the purpose of assisting the scalability of end-to-end multicast protocols. GMTS provides a rich set of filtering and aggregation functions that support feedback suppression and sub-tree multicast operations which are desirable for many multicast transport protocols. We describe the GMTS architecture, the set of services, and provide examples of how the services can be used to support reliable multicast, repair services, anycasting, and multicast congestion control.

## 1 Introduction

The development of scalable end-to-end multicast protocols poses a tremendous challenge to network protocol designers. For example the development of reliable multicast protocols has received considerable attention in recent years. Most protocols are based on an end-to-end solution [2,6,11] and have found the problem of scaling to 1000s or even 100s of receivers daunting. The primary obstacles to the development of scalable protocols have been *feedback implosion* and *transmission isolation*. The first of these concerns the difficulty for a large multicast application to limit feedback from receivers to a data source or to each other. The second concerns the difficulty of limiting the transmission of data to the subset of a multicast group that requires it.

There have been several proposals for adding functionality to routers for the purpose of improving the performance of multicast applications, particularly reliable multicast. Papadopoulos and Parulkar [8] introduced additional forwarding functionality to a router which would allow each router to identify a *special outgoing interface* over which to transmit a particular class of packets. They showed how this *turning point functionality* could be used to improve the performance of reliable multicast protocols. Levine and Garcia-Luna-Aceves [5] proposed the addition of *routing labels* to routing tables which could be used

to direct packets over specific interfaces. One of these, called a distance label, was shown to be quite useful in reliable multicast for directing requests for repairs to nearby repair servers. The third and, perhaps most relevant proposal is the PGM protocol [10]. Briefly, PGM is a reliable multicast protocol which uses negative acknowledgements (NACKs). The PGM protocol is an end-to-end transport protocol that contains a router component which performs NACK suppression and retransmission subcasting functionality. Our work is especially motivated by PGM and the recognition the utility in exporting a set of flexible, simple router-based functionality (such as was used in implementing PGM) for the purpose of protocol design. This would simplify the design of a large class of scalable multicast transport protocols.

In this paper, we present a set of Generic Multicast Transport Services (GMTS) that are intended to help protocol designers deal with these two problems. These services are designed to assist in the scaling of receiver feedback information and in providing subcasting services for large multicast groups. They consist of simple filtering and aggregation functions that reside within routers.

Signaling protocols are used from hosts to set up and invoke these services. Briefly, a session source first initializes one or more desired services on its multicast tree using GMTS setup messages. The GMTS-capable routers on the tree then aggregate feedback from receivers and/or isolate transmissions through the use of filters set by either the sender or the receivers. For robustness, periodic transmissions of setup messages on the multicast tree are used to refresh GMTS state in the face of routing changes and other possible errors. It should be stressed that GMTS services are only invoked for certain signaling packets; data packets are not treated any different and will not cause any additional processing in routers.

GMTS is not intended to provide sophisticated services which are difficult or impossible to implement in routers. GMTS services are implemented at the IP layer and provide unreliable *best-effort* services. Transport protocols which make use of GMTS must be robust in the face of failures and the absence of GMTS-capable routers in the network.

At a superficial level, GMTS resembles active networking. However, unlike active networking proposals, GMTS objects are simple and fixed (i.e. not dynamically uploadable modules). We feel that a small number of fixed services which, if made available, can benefit multicast transport protocols while, at the same time, are reasonable candidates for implementation in a router. Furthermore, GMTS objects are lightweight and contain only a small amount of state. This is in contrast to recently proposed active repair services that have been proposed by the active networking community, [3,4,9], which require the caching of packets for the purpose of providing retransmissions.

The paper is organized as follows. In the next section, we present a simple example of how GMTS can be used the context of a reliable multicast transport protocol. Section 3 introduces the generic transport services architecture. Section 4 describes a GMTS object, its state and methods, which is the fundamental building blocks for a GMTS session. Applications to the development of multicast

transport protocols are given in Section 5. The contributions of the paper are summarized in Section 6.

## 2 A GMTS Example: Reliable Multicast

Before describing the details of GMTS, we present a simple example in the context of a PGM-like reliable multicast protocol. A more detailed description of a reliable multicast protocol based on forward error correction (FEC) can be found in Section 5.1.

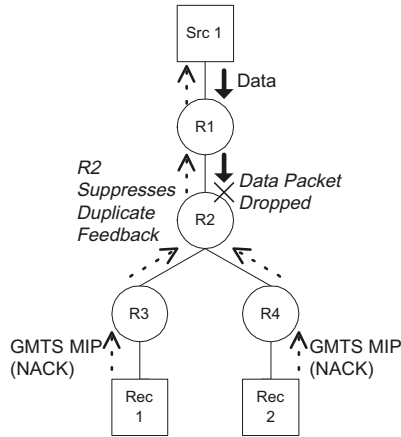
Consider a NACK-based reliable multicast protocol which places the responsibility of packet loss detection on each receiver. Each time that a receiver detects a loss (based on a gap in the sequence numbers of the packets that it receives), it unicasts a request for a repair (NAK) to the sender. Upon receipt of a NAK for a specific packet, the sender retransmits the packet to all receivers.

This protocol faces considerable challenges in dealing with multiple NAKs for the same packet. First, there is the problem of the sender having to process many NAKs. Second, there is the problem of limiting the number of retransmissions to the same packet. GMTS can be used to (partially) solve these two problems. Prior to the transfer of any data, the application sets up a NAK aggregation object at each GMTS-capable router using an setup messages. This object is set up to suppress NAKs for the same packet. In addition, the router maintains information regarding the interfaces over which it has received NAKs so that it can subcast the retransmission over the portion of the multicast tree that contains receivers requiring a retransmission of the packet.

In Figure 1, we show how GMTS can be used to aggregate feedback information in a reliable multicast transport protocol. In this figure, a multicast source (Src 1) is transmitting to two receivers (Rec 1 and Rec 2). The data packets from Src 1 are treated as regular multicast packets and forwarded accordingly. On the link between router R1 and router R2, a data packet is lost. Assuming a NAK based reliable multicast protocol, this loss will cause the receivers to send a NAK to the source for the data that was lost. In the example, receivers use GMTS to send the feedback (i.e. the NAKs) to the source. GMTS router R2 treats these NAKs in a special manner, suppressing the redundant NAKs to the source. Therefore, only one NAK arrives at the source. We can see from this example that GMTS routers only certain types of packets require additional processing at GMTS routers and that the majority of end-to-end packets are forwarded according to normal multicast forwarding rules (i.e. without additional router processing).

## 3 Generic Multicast Transport Services

A GMTS session is instantiated by a multicast sender and operates over (unreliable) IP multicast. A GMTS session, identified by source and group address, consists of objects located in GMTS-capable routers on the multicast tree connecting the source to the receivers. A set of signaling protocols serve as the



**Fig. 1.** Example of network support for transport protocols.

interface to GMTS objects and are used to initiate objects and to invoke object methods remotely. If a multicast application contains more than one sender, a GMTS session is established for each of them.

GMTS methods are only invoked for certain types of signaling packets generated by an end-to-end multicast protocol; most data packets are sent end-to-end as regular multicast packets and are not treated specially by GMTS-capable routers. For example, in a reliable multicast transport protocol, NAKs would be trapped by GMTS-capable routers while regular data packets would flow through the normal router forwarding path. We stress this to show that GMTS does not incur significant overhead in multicast routers as only periodic signaling packets are specially processed.

In this section, we provide a high-level description of the objects, the signaling mechanisms, and illustrate how these components can be combined through a simple example.

### 3.1 GMTS Objects

Many GMTS services (i.e. feedback suppression) require that routers keep a certain amount of state in order to provide those services. The relationship between services and the state required for these services is analogous to the relationship between object-oriented classes and their methods. It is for this reason that we use object-oriented programming terminology to define a GMTS object. A GMTS object is a set of methods (i.e. the actual GMTS services) and their associated supporting state which exist in routers on a multicast tree.

GMTS objects are the fundamental components which provide a fixed set of simple services through their methods. GMTS objects consist of state-dependent filtering methods, the state required for these filters, and methods for modifying this state. GMTS objects are very flexible in that they can support a rich set of

filters and state manipulation functions. In the paper, we illustrate the flexibility of these objects by illustrating their use in supporting an FEC-based reliable multicast protocol similar to PGM [10], end-host based repair services, multicast congestion control, and anycast.

GMTS objects are instantiations of available GMTS object types. A GMTS object consists of state variables and several methods that can be remotely invoked by either a source or a receiver. GMTS object types are predefined; that is, they are fixed specifications that are implemented in router software. The primary goal of these object types is to be able to set up and tear down simple filtering mechanisms that can be used by the routers to reduce the amount of end-to-end control traffic generated within a multicast session. GMTS objects are the actual per session instantiations of available GMTS object types.

GMTS objects and their methods are accessed via two signaling mechanisms. The first is the mechanism used to set up and refresh GMTS object state. State set up packets (SSPs) are used to set up objects and refresh their state. GMTS method invocation packets (MIPs) are used to invoke GMTS methods.

Because of our adherence to the soft-state philosophy, a timer is associated with each object. This timer is set at the time that the object is first initialized (i.e., the time that the router first receives a set up packet (SSP) listing that object), and reset each time that the router receives an additional SSP listing that object. If the timer expires prior to the receipt of a new set up packet, the object is deleted at the router. GMTS objects may have additional soft-state timers as required by the object. For example a NAK suppression object, as might be used in a reliable multicast protocol, would maintain a timer with the state associated with a particular sequence number.

GMTS object methods may be invoked on the multicast tree in either a reliable or an unreliable manner. Each set up packet contains a list of objects for which the sender wishes to instantiate (or to refresh the state for). Optionally, a sender may list options for methods specifying which methods will require reliable or unreliable delivery. For a reliable method, a GMTS router will verify the receipt of a MIP to the next hop GMTS router. For an unreliable method, a router will only transmit the MIP once to next hop GMTS router. Details can be found in [1].

GMTS object methods also include access rights for each method stating whether the sender, receiver, or both may invoke a given method. The GMTS sender may set access rights to all objects.

### 3.2 GMTS State Setup Packets

A state setup packet (SSP) declares objects to be set up in GMTS capable routers along a multicast forwarding tree. An SSP is multicast by the sender to its multicast group. As it traverses the multicast distribution tree, it is trapped by all GMTS-capable routers. Each router then creates objects corresponding to the object type declared within the SSP. An SSP can also be used to refresh an object, as will be discussed later, and to describe particular options for objects.

The second purpose of the SSP is to inform a GMTS router of the address of the next upstream GMTS router. This is used to allow GMTS sessions to traverse regions of non-GMTS capable routers. We define the GMTS tree as the tree of GMTS capable routers along the multicast forwarding tree.

GMTS must be robust in the face of network topology changes. Another use of a SSP is to refresh GMTS state and to re-instantiate it when the network topology changes. SSPs are sent periodically so that, in the event of a routing change, GMTS state will be set up along the new multicast routing tree.

### 3.3 GMTS Method Invocation Packet

The sender and receivers use a second signaling mechanism to remotely invoke methods on objects residing in GMTS routers. It consists of the transfer of method invocation packets (MIPs) which identify the methods to be invoked along with their required parameters. In addition, a MIP may include actual end-to-end data.

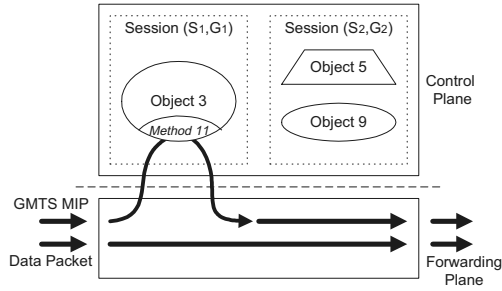
MIPs are primarily used by receivers to invoke object methods. These methods typically perform some kind of state or control message aggregation. Senders may also invoke methods. For example, a receiver might invoke a method to suppress NAKs during a reliable multicast session. A sender might invoke a method in order to change the timeout value of a GMTS object. Methods flowing toward receivers are always multicast. Methods flowing towards the source are always unicast between GMTS-capable routers.

The particular end-system protocol using GMTS needs to map GMTS services into its own set of variables. GMTS routers have no knowledge of the function of a particular piece of data on which a method is invoked. In order to provide an agnostic view of an end-system protocol to routers, we introduce GMTS *identifiers*. A GMTS identifier is associated with a particular object instantiation. It is used by an end-system protocol to map protocol specific data to GMTS objects. For example, a NAK based reliable multicast protocol may create an identifier for its sequence number space.

To a GMTS capable router, an identifier associates an incoming packet with an object. MIP packets contain both an identifier and a method. A router then looks up the object using the identifier and applies the particular method to the packet. In order to make GMTS available to all types of protocols, routers have no knowledge of the mapping between objects and end-system protocol parameters. The source of GMTS SSP packets creates identifiers for each object that it wishes to create.

Figure 2 illustrates the behavior of a GMTS router when presented with a MIP and with a non-GMTS packet. The first packet is a GMTS MIP, which causes a router to perform special processing on the packet. This packet, for example, would contain end-to-end protocol signaling or would be a special data packet with special forwarding rules (e.g. subcasting). The second packet is a regular multicast data packet which was sourced from either a non-GMTS session or GMTS session. As stated earlier, GMTS sessions use regular multicast packets





**Fig. 2.** Paths that data packets and MIPs take through a GMTS router.

for the bulk of their data. These packets are forwarded normally, and without latency cost according to multicast forwarding rules.

We show two GMTS sessions within a router, identified by their tree forwarding entries. The first session, identified by the  $(S_1, G_1)$  forwarding entry has one object with identifier three. As the GMTS MIP packet traverses the router, it invokes a particular method (method 11) on object 3. In most cases, the MIP would contain signaling feedback or be a special data packet to be subcasted. In the second session, two objects of differing types are shown; in this case, a host is using services provided from two different object types.

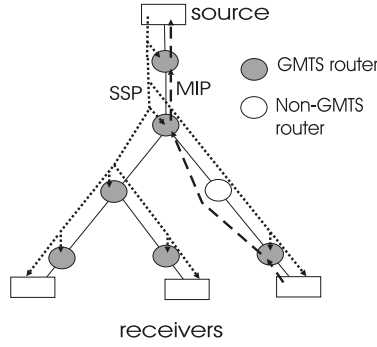
### 3.4 GMTS Operation

The following describes the steps required in setting up a GMTS session:

1. The transport protocol designer decides which services will be needed by the protocol. GMTS services are chosen to achieve end-to-end scalable signaling for a multicast transport protocol. The services required of the end-to-end protocol dictate what objects to set up.
2. The transport protocol sender multicasts GMTS SSPs to instantiate the objects desired by the protocol (Figure 2). An SSP contains a list of all objects desired, their types, identifiers, and timeout values.
3. SSPs are periodically issued to refresh objects.
4. SSPs inform receivers of the identifiers to use for the instantiated objects
5. Receivers (or senders) issue MIPs which cause methods to be invoked on the multicast tree. MIPs contain an object identifier and the method ID that they wish to invoke. Figure 2 illustrates the rightmost receiver sending a MIP towards the source. Observe that it is unicast between GMTS routers.

## 4 GMTS General Purpose Object (GPO)

In this section we describe a general purpose GMTS object (GPO) type that provides a rich set of services to end-system multicast protocols. The GPO contains several methods providing a set of core services useful to protocol designers and are reasonable for implementation in routers. These include:



**Fig. 3.** Operation of GMTS.

**Table 1.** Private variables for the GMTS General Purpose Object.

Private Variable Name	Description
<i>start_window</i> , <i>end_window</i>	Lowest and highest allowable sequence numbers on which methods may be invoked.
<i>state</i>	<i>state(i)</i> identifies the sequence state for sequence number <i>i</i> .
<i>interface_vector(i)</i>	<i>interface_vector(i)</i> is vector of state for all interfaces marked for sequence number <i>i</i> .
<i>object_expiration_timer</i> , <i>state_expiration_timer</i>	Timers for object and sequence state

- Suppression: simple feedback suppression towards a source.
- Predicate Suppression: suppression of redundant feedback based on the boolean result of one or more comparison operations.
- Subcasting: the ability to forward a packet to a subset of the multicast forwarding tree.

#### 4.1 Sequence Space

Many end-to-end multicast protocols use a sequence space. In order to apply suppression type operations per sequence state (i.e. NAK suppression), the GPO includes a large sequence number space which can be used to identify a set of state. Sequence numbers can represent many different parameters in multicast transport protocols. This sequence number is used to reference all state in the GPO. The private variables associated with the GPO are listed in Table 1.

*start\_window* and *end\_window* specify a range of sequence numbers which are currently valid and for which state is kept. Associated with sequence number *i* are two state variables. The first, *state(i)* is a nonnegative integer. The second, *interface\_vector(i)* is a vector which contains as many integer valued components as outgoing interfaces at the router, including the one towards the

sender. As we will observe shortly, this vector is used to determine the interfaces over which receiver and sender initiated methods are to be invoked.

## 4.2 Methods

Methods are used for state maintenance and the actual GMTS services. GPO methods are listed in Table 2 and described briefly below.

The *modify\_window* method to update the GPO sequence number window. All state associated with sequence numbers outside the new window is discarded. *clear\_state*, is used to clear state within the range defined by *start\_range* and *end\_range*. *modify\_state*, explicitly sets the state associated with a sequence number.

Consider *rcvr\_update*( $n, v, pred, f_s, f_v$ ). If  $pred(v, state(n))$  is true, then  $state(n) := f_s(v, state(n))$  and  $interface\_vector(n) = f_v(v, interface\_vector(n), vec)$  where *vec* is the vector of all 0s except for a one in the position corresponding to the interface over which the method was invoked, If  $state(n)$  has changed, then the *rcvr\_update* method is invoked on the link directed towards the source with  $v = state(n)$ , and all other arguments unchanged. Otherwise, nothing occurs. It is expected that a receiver will normally invoke this method.

Method *mcast\_update*( $n, v, pred, f_s, f_v$ ), behaves in a similar manner except that, whenever the predicate is true, the method is invoked on all outgoing links that are part of the multicast tree except the one that it arrived over. Note that the MIP containing *mcast\_update* is multicast over all outgoing links in the routing table and unicast on the outgoing link on the path to the next upstream GMTS router on the path towards the source associated with the GMTS session.

Last, *forward*( $n, v, g_s, g_v, data$ ) results in the invocation of the same method on the routers attached to all outgoing interfaces (except the one that the original invocation arrived on) for which the corresponding components in  $interface\_vector(n)$  are greater than  $v$ . In addition,  $state(n) := g_s(v, state(n))$  and  $interface\_vector(n) := g_v(v, interface\_vector(n))$ . Upon receipt by a receiver, the parameters *data*, *state*, and *n* are delivered to the application.

In addition, there are timers associated with each individual state component and with the object itself. *state\_expiration\_timer*, and *object\_expiration\_timer* contain the values they are set to when initialized. They can be reset using *set\_object\_timer* and *set\_state\_timer*.

## 4.3 Predicates and Operations

A number of methods require the specification of a predicate used to determine if state should be modified. If the application of the predicate is true, the given operations are performed. The functions  $f_s()$  and  $f_v()$  are used to modify the sequence state and the interface vector function. Examples can be found in [1].

**Table 2.** Methods for a GPO.

Method Name	Arguments	Method Description
<i>modify_window</i>	<i>start, end</i>	Modify sequence window
<i>rcvr_update</i>	<i>n, v, pred, f<sub>s</sub>, f<sub>v</sub></i>	Usually invoked by receiver to invoke suppression or election based on predicate <i>pred</i> . If TRUE, set state with <i>f<sub>s</sub></i> () and interface vector by <i>f<sub>v</sub></i> ().
<i>clear_state</i>	<i>start, end</i>	Clears all sequence state in the provided range.
<i>forward</i>	<i>n, v, g<sub>s</sub>, g<sub>v</sub>, data</i>	Usually invoked by sender to allow transmission of <i>data</i> on interface <i>k</i> such that <i>interfaces_vector(k) - value &gt; 0</i> . State and interface vectors set using <i>g<sub>s</sub></i> () and <i>g<sub>v</sub></i> (). <i>data</i> is included in the MIP.
<i>mcast_update</i>	<i>n, v, pred, f<sub>s</sub>, f<sub>v</sub></i>	Similar to the <i>rcvr_update</i> method except MIP is multicast on all interfaces on tree except one it arrived on.
<i>modify_object_timer</i>	<i>v</i>	Set the object timer to <i>timer_value</i> .
<i>modify_state_timer</i>	<i>n, v</i>	Set the sequence number state timer to <i>timer_value</i> .

## 5 Examples

We present two examples of protocols constructed using the general purpose object. These are a receiver oriented reliable multicast protocol using forward error correction (FEC) and a protocol that can aid in providing scalable congestion control to a multicast session. Other examples are found in [1].

### 5.1 A Reliable Multicast Protocol with FEC

FEC has been shown to be especially effective in the implementation of reliable multicast, [7]. Consequently, in this section we describe a receiver oriented protocol that uses FEC to reduce the number of retransmission requests. The design is similar to that of PGM [10]. However, the purpose of this exercise is not to develop new protocol but to illustrate the flexibility of GMTS.

Briefly, data is grouped into blocks consisting of  $B_{size}$  packets. Each time that a receiver detects that it has not received all of the packets in a particular block, it forwards a request for a number of parity packets equal to the number of missing packets. Upon receipt of this request, a GMTS-capable router checks to see if an earlier request for at least as many parity packets has already passed through for this data block. If so, the parity request (PR) is discarded. If there has been no previous request for as many parity packets, then the request is forwarded towards the source. In either case, a record is made of the number of parity packets required to be sent down the interface over which the parity request arrived.

The source creates parity packets in response to a parity request and sends them down the tree to the receivers. Each GMTS router sends as many parity packets over each outgoing interface as has been recorded for that block.

This protocol uses a single object at each router to perform feedback suppression and parity transmission subcasting. The sender maintains the following state:

1. Block sequence number window  $(n_{start}, n_{end})$ ; this corresponds to the blocks which the sender is prepared to create parity packets for.
2. Sequence number for the next block to be transmitted,  $n_{next}$ .
3. Timer for clocking SSPs.
4. Maximum parity sequence number for block  $j$ ,  $P_{max}(j)$  corresponding to maximum number of parities requested for block  $j$ ; initially set to zero.

Each receiver maintains

1. Block sequence numbers of packets received in last previously multicast sequence number window.
2. Parity request (PR) suppression timer for block  $j$ .  $T_{pr}(j)$ ,
3. PR loss detection timer for block  $j$ ,  $T_{loss}(j)$ .
4. Number of parities required to recover missing packets belonging to block  $j$ ,  $R_{rr}(j)$ .

**Initialization:** The source periodically multicasts an SSP which declares a GPO, its timeout, and the start and end of the sequence window.

**Data transfer:** Data packets are multicast in blocks of size  $B_{size}$  to the group. Unique sequence numbers commencing with  $n_{start}$  are used to identify the blocks. They are incremented at the transmission of each new block. In addition, each packet has a unique sequence number in the range  $(0, B_{size} - 1)$ . The receivers use these sequence numbers to identify packets unsuccessfully received for each block. Last, these packets require no GMTS support. Therefore, they are transmitted as non-GMTS packets. They traverse the fast path illustrated in Figure 2.

**Repair requests:** When a receiver detects a loss within block  $j$ , it sets a parity request counter  $R_{pr}(j)$  to the number of missing packets and the PR suppression timer. If the timer goes off, the receiver increments  $R_{pr}(j)$  by the number of additional packets missing from block  $j$  and invokes *rcvr\_update* with  $v = R_{pr}(j)$ ,  $f_s(x, y) = x$ , and  $f_v(a, v_1, v_2) = \max(v_1, a * v_2)$  at the first upstream GMTS router. The PR loss detection timer is also set. If the missing packets for block  $j$  or a PR for block  $j$  containing a value greater than  $R_{pr}(j)$  arrive while the suppression timer is set, the timer is turned off. In the case of receipt of a repair request, the RR loss detection timer is still set. Last, if the PR loss detection timer goes off, then the process is repeated.

**Parity transmission:** When the sender receives a PR packet for block  $j$  requesting  $v$  parities where  $v > P_{max}(j)$ , it constructs  $v - P_{max}(j)$  new parity packets for block  $j$  and invokes the forward method to send them towards the receivers. The sender updates  $P_{max}(j)$  to  $v$  and invokes the forward method with  $n = j$ ,  $v = 0$ ,  $g_s = x - 1$ ,  $g_v = interface\_vectors(n)$ , and the appropriate parity packet as parameters.

Variations of this protocol are described in [1]

## 5.2 Congestion Control

Several congestion control protocols require knowledge of the worst case receiver according to some metric such as loss rate,  $p$ . This is easily accommodated in the GMTS framework by setting up a suppression object and having the receivers periodically invoke the *rcvr\_update* method up the tree. The largest receiver loss rate would propagate up the tree and the routers would establish a path between the source and the receiver that generated this value.

## 6 Summary and Future Work

GMTS provides an architecture for the development of simple services in routers that assist control message scalability problems in end-to-end multicast protocols. We have shown how GMTS can be used to complement an end-to-end reliable multicast protocol.

We continue to investigate further services that may be useful to a variety of multicast protocols and applications. Of particular interest is the area of congestion control and RTCP like reporting functions. Last, we are in the process of adding GMTS to a Linux-based router for the purpose of evaluating its performance and the benefits provided to multicast applications.

## References

1. B. Cain, D. Towsley. "Generic Multicast Transport Services: Router Support for Multicast Applications," UMass Computer Department Technical Report TR99-74, Nov. 1999.
2. S. Floyd, V. Jacobson, S. McCanne, C. Lin, L. Zhang. "A Reliable Multicast Framework for Light-weight Sessions and Application Level Framing", *IEEE/ACM Trans. on Networking*, **5**, pp. 784-803, Dec. 1997.
3. S. Kasera, J. Kurose, D. Towsley. "A Comparison of Server-Based and Receiver-Based Local Recovery Approaches for Scalable Reliable Multicast", *Proc. INFOCOM'98*, 1998.
4. L.H. Lehman, S.J. Garland, D. L. Tennenhouse. "Active Reliable Multicast", *Proc. INFOCOM'98*, 1998.
5. B.N. Levine, J.J. Garcia-Luna-Aceves. "Improving Internet Multicast with Routing Labels", *Proc. ICNP-97*, pp. 241-250, Oct. 1997.
6. J. Lin, S. Paul. "RMTP: A reliable multicast transport protocol", *Proc. of IEEE INFOCOM'95*, 1995.
7. J. Nonnenmacher, E. Biersack, D. Towsley. "Parity-Based Loss Recovery for Reliable Multicast Transmission", *IEEE/ACM Trans. on Networking*, Aug. 1998.
8. C. Papadoulos, G. Parulkar. "An Error Control Scheme for Large-Scale Multicast Applications", *Proc. INFOCOM'98*.
9. D. Rubenstein, S. Kasera, D. Towsley, J. Kurose. "Improving Reliable Multicast Using Active Parity Encoding Services (APES)", *Proc. INFOCOM'99*.
10. T. Speakman, D. Farinacci, S. Lin, A. Tweedly. "PGM Reliable Transport Protocol", IETF <draft-speakman-pgm-spec-02.txt>, August, 1998.
11. D. Towsley, J. Kurose, S. Pingali. "A comparison of sender-initiated and receiver-initiated reliable multicast protocols" *IEEE JSAC*, April 1997.

# RP-Based Multicast Receiver Access Control in PIM-SM

Thomas Hardjono

Nortel Networks  
600 Technology Park Drive  
Billerica, MA 01821, USA,  
thardjono@baynetworks.com

**Abstract.** The current work focuses on the issue of *receiver access control* in the context of the *Protocol Independent Multicast* (PIM) protocol. Currently, a host within a subnet can request the multicast router to join any multicast group without that host being authenticated and authorized to join. This (unauthorized) join-request results in the multicast distribution tree being extended towards that subnet, which opens the possibility of attacks. In such an attack, the malicious user/host intentionally extends or “pulls” the tree towards its subnet, effecting a wastage in resources and state within all the affected routers. In this case, the end-to-end encryption of the multicast data does not provide any help, since the (encrypted) packets still flows down the distribution tree to the malicious host. The current work analyzes this problem closer in the context of PIM Sparse Mode (PIM-SM) and offers a solution. The proposed approach also complements the recent developments in IGMPv3 [1] and the Express multicast model of [2].

## 1 Introduction

IP multicast is emerging to be the future vehicle of delivery for multimedia in the Internet, with the promise of reaching the millions of users on the Internet. One crucial architecture component to this future vision is the multicast routing protocol that delivers multicast data packets (data stream) to group members, following the basic IP multicast model proposed in [3].

A number of multicast routing protocols have been proposed in the last few years (eg. [4–7]). However, one protocol that has the promise of emerging as the industry standard in the *Protocol Independent Multicast* (PIM) multicast routing protocol, of which a *dense* mode (PIM-DM) and a *sparse* mode (PIM-SM) have been defined [7]. Currently, PIM has gone through over two years of development and experiments, and a number of large *Internet Service Providers* (ISP) have began to enable PIM in their routers.

In this paper we focus on the issue of receiver access control to the multicast distribution tree within the context of PIM-SM. Currently no mechanism exists within PIM-SM to control the joining of hosts to the multicast distribution tree emanating from the *Rendezvous Point* (RP). A host can execute a group

membership protocol and direct its subnet router to join a multicast group, regardless of whether or not the host is an actual member of the multicast group. Encryption of data – typically end-to-end – does not protect against the malicious host from joining the group (and discarding packets). In itself, the result of the distribution tree being extended to the host’s subnet has already consumed bandwidth, which in this case is simply wasted and thus leads to the potential of a denial-of-service attack on other hosts in the domain.

In the following section we briefly discuss the PIM-SM protocol (Section 2), followed by a problem description in Section 3. Section 4 reviews existing work directly relevant to the issue at hand, while Section 5 suggests a solution to the problem based on the use of the RP and the group-key management protocol. The paper is then closed with some remarks and pointers for future work. The notations employed in the current work is as follows. Public key pairs are denoted as  $(SK, PK)$  representing the Secret-Key and the Public-Key, whilst private/symmetric keys are denoted by the symbol  $K$ . The reader is assumed to be familiar with the PIM-SM protocol, and is directed to [7] for more information.

## 2 Context: The PIM-SM Protocol

The *Protocol Independent Multicast* (PIM) protocol is a multicast routing protocol designed with a number of motivations, one being the scalability of the protocol. To this end, two related modes of the PIM protocol have been defined, namely the PIM dense mode (PIM-DM) protocol and the PIM sparse mode (PIM-SM) protocol.

PIM-DM, is aimed at domains or regions whose group-population is dense and thus warrants the use of flood-and-prune techniques similar to that in DVMRP [4]. However, unlike DVMRP and MOSPF [6], PIM is “protocol independent”, meaning that PIM does not depend on any specific unicast routing protocol/table (as do DVMRP and MOSPF).

PIM-SM, on the other hand, was designed for network with a sparse group-population in which techniques such as flooding could not be justified from a bandwidth point of view. Thus, in PIM-SM a number of special “meeting-point” routers, called *Rendezvous Point* (RP), are designated to which the receivers’ join requests are directed (Figure 1). Before any groups can function, a *Bootstrap Router* (BSR), must advertise the set of (candidate) RPs which are available in the domain. Following this meeting-point philosophy, a sender (Source) wishing to multicast data to the group initially sends data messages for the group via unicast (encapsulated) to the RP. The RP forwards the data to the receivers using a unidirectional shared tree. In order to avoid the RP becoming a bottleneck for high data-volume groups, when the sender’s traffic exceeds a pre-defined threshold a receiver (ie. its DR) must establish a shortest path tree (SPT) between itself and the Source. Thus, at this point, the intermediate routers (between the RP and the receivers) must also “switch” to the shortest path tree rooted at the Source.



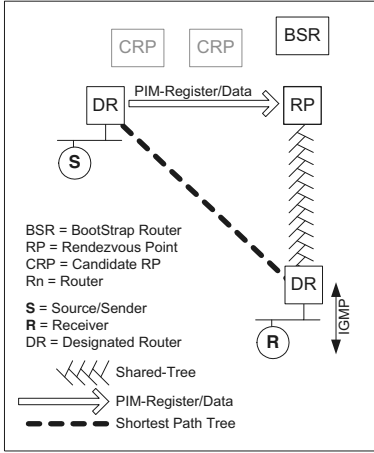


Fig. 1. PIM Sparse Mode

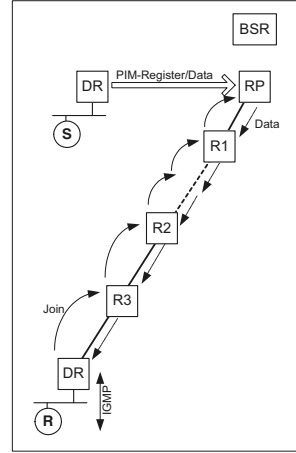


Fig. 2. Unauthorized “pulling”

### 3 Access Control for Receivers: Problem Statement

One of the main factors that makes IP multicast attractive from the perspective of scalability is the anonymous-receiver model underlying it. Any host in a subnet can join a multicast group without its subnet multicast router passing identification information about the host to other routers upstream in the distribution tree. This allows IP multicast to scale to a large number of participating hosts. However, from the perspective of security, this lack of host-identification information represents a problem for access control. This situation is true for most multicast routing protocols in the industry today, including PIM.

A possible attack that exploits the anonymous-receiver underpinnings of IP multicast is one in which a host simply joins a multicast group, without any intention of using the data being delivered to it. In such an attack, the user/host essentially extends or “pulls” the tree towards the subnet effecting a wastage in resources and state within all the affected routers. In this case, the encryption of the multicast data does not provide any help, since the (encrypted) packets still flows down the distribution tree to the malicious host.

The work of [8] has attempted to increase the security of the PIM protocol by requiring all control-messages to be authenticated via IPsec AH and a symmetric key shared by all routers in the PIM domain (namely, the “equal-opportunity-key”,  $K_{eq}$ ). However, this approach does not solve the main concern here of controlling access to the multicast distribution tree.

More specifically, although the control messages between a designated router (DR) and the RP are per-hop authenticated, the interaction between the Receiver and its DR remain open to attack. Thus, using the example in Figure 2, an unauthorized Receiver R is still able to join a multicast group, even though the control-message that flow upstream (from its DR to router R3, upwards to R1 and finally the RP) is protected from any malicious modifications via the equal-

opportunity-key. The RP will still respond in the usual manner by sending out data along the appropriate interfaces towards the subnet of the unauthorized Receiver R.

## 4 Related Work

The main effort to protect the PIM distribution tree has been put forward in [8], with a key management proposal given in [9]. Other related work is the recent IGMPv3 [1] which allows a filtering of Sources, and the *Express* multicast proposal of [2] which proposes joins using specific (Source, Group) pairs. These are reviewed briefly in the following.

### 4.1 Control Message Authentication in PIMv2

The PIM Working Group (IETF) has recently put forward a proposal in [8] for the arrangement of cryptographic keys within a given PIM domain, with the aim of deploying the keys for control-packet authentication in the PIM domain. Following [8], when security is enabled, all PIM version 2 messages will carry an IPsec authentication header (AH) [10]. The authentication mechanism must support, among others, HMAC-MD5-96 [11, 12] and HMAC-SHA1-96 [13] security transformations.

The PIM key arrangement of [8] identifies the following entities in a PIM domain that require keys: the *Bootstrap Router* (BSR), the *Rendezvous Point* (RP), the *Designated Router* (DR) and other PIM routers. All keys are relevant and recognized only within one PIM domain. The keys are as follows:

- *BSR Public Key*: All BSRs own an identical RSA [14] key pair<sup>1</sup> and uses the private key to sign an entire bootstrap message, whilst other PIM-routers only have the public key to verify the signature. This RSA secret-public key pair is denoted here as  $(SK_{bsr}, PK_{bsr})$ . This allows only authorized candidate BSRs to become a bootstrap router.
- *Equal Opportunity Key*: All PIM routers in the same domain share a single private (symmetric) key used to compute digests or MACs for the protection of PIM control messages. This key is denoted here as  $K_{eq}$ . This key is used for per-hop authentication of control messages by PIM-routers in a given PIM-domain.
- *RP-Key*: All RPs and BSRs share another private (symmetric) key, known as the “RP-key” and denoted here as  $K_{rp}$ . No other routers have this key. For candidate RP advertisement the digest is only calculated with the RP-key  $K_{rp}$  (instead of the equal opportunity key  $K_{eq}$ ). This achieves the effect that only the authorized candidate RPs can advertise their candidacy to the BSR.

---

<sup>1</sup> Although the first version of the proposal in 1998 required one RSA key pair to be shared by all BSRs, the second revision of the proposal (IETF-45, July 1999) recognizes the need for each BSR to have a unique RSA key pair.

## 4.2 IGMPv3 and Express

The most recent version of IGMP (ie. Version 3 [1]) provides a receiver with the ability to filter based on specified sources. Thus, in the interface provided to the receiver, the receiver must not only specify the multicast-address (ie. group) which it wishes to join, but also specify the source-list and the filter-mode. From a multicast perspective, this addition of source-based filtering represents a major step forward as it allows the controlling of the senders to a multicast group. The work of [1] recognizes a number of possible control-message forgeries (eg. the IGMP Query message and Report messages), thereby recognizing the need of user/host authentication in IGMP and in IP multicast.

Along similar lines, a departure from the basic IP multicast service model was recently proposed in the form of the *Express* model [2]. Here, a “multicast channel” is defined to consists of the tuple  $(S, E)$  where  $S$  is the sender’s source address and  $E$  is the channel destination address. The receiver or subscriber then requests reception of the data sent to the channel by explicitly specifying both  $S$  and  $E$ . The work of [2] defines a number of interfaces to the source and subscriber, where a source uses  $channelKey(channel, K_{(S,E)})$  while a subscriber uses  $newSubscription(channel, K_{(S,E)})$ .

As the effort of [2] does not specify authentication methods or key delivery and management approaches to be deployed, we see the current work as complementing the works of [2] and [1]. Indeed, the proposal to be put forward in the following section provides a way for the key  $K_{(S,E)}$  of [2] to be delivered to both the source and subscribers of a channel.

## 5 Coupling with Group Key Management

One major advantage of the PIM protocol from the point of view of access control lies in the fact that the Rendezvous Point (RP) is initially the point of departure for data packets destined for the multicast group members. The Source (Sender) unicasts data packets to the RP encapsulated within the PIM “Register” messages. Thus, the RP presents itself as a suitable point at which a decision is made as to whether data is sent towards a (candidate) Receiver, effecting the creation of a branch within the multicast distribution tree, linking the Receiver and the RP. (Note that the RP also represent a good location to conduct Source/Sender access control, since sources within the PIM protocol must initially unicast data to the RP).

In this section we propose the use of the group key management (GKM) event to aid the RP is deciding access control. We introduce a security entity called the *Domain Key Distributor* (DKD) which has a number of roles in the domain, one of which is to be a key-server for the domain [15]. The DKD also acts as a public key certificate for open public keys (namely public keys known outside the PIM domain) and “closed” public keys (only known within the domain). The DKD also plays an important role in the key management within the domain. Although functionally it is referred to as a single entity, in practice the DKD can

be implemented by several servers for reliability and availability reasons. Any such server must be implemented with the strongest security protection available due to their sensitivity.

### 5.1 Group key management: background

Since data related to a multicast group traverses the public Internet and is therefore subject to tapping or copying by non-members of the group, encryption is the method commonly used to provide control to the data. In the simplest case, shared-key (symmetric) cryptography is used by the Sender/Source and the Receivers, where the data is encrypted by the Sender and decrypted by the Receivers. This shared-key is commonly referred to as the *group-key*, since only members of the multicast group are in possession of the key.

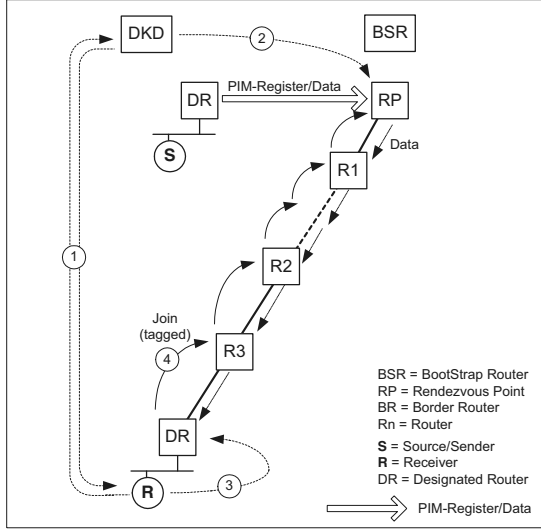
The use of cryptography necessitates the delivery or dissemination of keys, which in this case is the group-key. Thus, an additional facet to the general problem of multicast security is the method of distributing keys to the appropriate entities involved in a multicast instance and the management of the keys of over given period of time. A *Group-Key Management* (GKM) protocol must not only issue a group-key for a new multicast group, but also update (re-key) the existing group-key under certain conditions and following the prescribed policies, be those general security policies or multicast-specific policies. A GKM protocol must be scalable, secure and must be independent from the underlying unicast and multicast routing protocols [16]. Examples of GKM protocols are [17–19].

It is important to note that the use of a group-key on multicast data is carried-out *end-to-end*, independent of the multicast routing protocol. The routing protocol – in this case PIM – is unaware of the group-key, and it treats the (encrypted) data simply as payload. Neither the RP nor the other PIM entities (eg. routers) understand the notion of the group-key. The encryption of data by a valid Sender (destined for other valid group members that hold a copy of the group-key) does not protect against the multicast distribution tree being attacked by unauthorized receivers.

### 5.2 GKM event to facilitate receiver access control

Since a new member of a multicast group must use the GKM protocol (designated for that group) in order to obtain a copy of the current group-key, we propose to use that event to also perform receiver access control with the aid of the RP and the DKD. This is further illustrated in Figure 3.

In Figure 3 (Step 1), the receiver R joins the multicast group by first obtaining a copy of the group-key (and other parameters) from the Domain Key Distributor (DKD). User/host authentication is carried-out at this time (eg. based on certificates), and group membership is also verified. This behaviour represents the general approach taken by most – if not all – GKM protocols. In the current context, we introduce an additional cryptographic key called the “DR Key” (*DR-key*), denoted as  $K_{dr}$  for the purpose of allowing the RP to



**Fig. 3.** GKM Event for Receiver Access Control

determine whether a DR in a given subnet is catering for a valid member of a group (as opposed to an unknown member or an attacker).

Here we propose that when a host-member is authenticated/verified by the DKD and is given a copy of the group-key, that host-member is also given a new DR-key (in addition to other parameters supporting its usage). The DR-key is only ever used once, and the DKD provides a list of the current DR-keys to the the RP through a secure channel (Step 2). The host-member then delivers the DR-key to its subnet-router (ie. the DR) through a secure channel, such as a secure unicast protected using IPsec ESP (Step 3). Extensions to IGMP can also be created to cater for this function.

When the DR issues the Join message towards the RP, it must create a “tag”, which is a digest/MAC computed using a keyed-hash function and the DR-key. In addition, a nonce is also added to the Join message to prevent replay attacks. Other replay-prevention techniques can also be deployed. The triple (*Join*, *Tag*, *Nonce*) is then treated as payload, to be protected using the mechanisms specified in [8], namely using IPsec AH and the equal-opportunity-key ( $K_{eq}$ ). The tagged-join message is then sent towards the RP by the DR (Step 4).

Upon receiving the tagged-join message, the RP uses its copy of the DR-key to verify the received join message. If the verification fails, the message is dropped and the relevant routers along the path between the DR and RP will time-out due to the fact that no data packets are received from the RP. If the verification is successful, the RP will deliver data to the appropriate interface, following the usual PIM protocol.

Note that in this proposed solution, the routers within the multicast distribution tree do not maintain any host-identification information. Hence, the solution still promotes the “anonymous-receiver” approach underlying the IP multicast model of [3].

### 5.3 Features and advantages of the basic solution

The solution described above has a number of features which makes it attractive:

- *Independence of group key management.* The solution maintains the important principle of the independence of group key management from the underlying multicast routing [16]. Multicast routing is unaffected by the introduction of the DR-key, since the key is introduced at the RP and the DR.
- *Adherence to the basic IP multicast model.* The solution adheres to the basic IP multicast model of [3], where the receivers are still anonymous from the point of view of the multicast distribution tree. The proposed approach also complements the recent developments in IGMPv3 [1] and the Express multicast model of [2].
- *Deploys existing GKM protocols.* The solution does not require a new GKM protocol, but only the introduction of additional security parameters to the existing deployed GKM protocol.
- *Adherence to the key arrangement of [8].* The solution fits into the key arrangement proposal for PIM and strengthens the security of PIM-SM as it currently stands.

### 5.4 Join towards on-tree nodes

The approach summarized above is deployable as it stands for new multicast distribution subtrees that emanate directly from the RP. Further refinement is required to cater for the situation where the join (tagged) command hits upon an on-tree node/router, representing the root of a sub-tree.

To refine the proposed basic solution in Section 5.2, consider the scenario in Figure 4. In Figure 4 (a) a Receiver R2 requests its designated router DR2 to join a multicast group through IGMP. Data is already flowing down from the RP to DR1 at the left-hand subtree (via Routers R1, R2, R3 and finally to DR1). In this scenario, Router R2 is the branching-router.

When a router receives a tagged-join message it must verify whether or not it is a branch-router (eg. by checking whether or not it already has an outgoing-interface for the same group G). If it does not have any outgoing-interfaces, then the router concludes that it is not on the distribution tree for that multicast group G.

However, if the router discovers that it is already an on-tree node within the multicast distribution tree for group G, it must perform additional tasks beyond those defined for PIM-SM. Rather than simply forwarding data to another interface, it must wait for an explicit acknowledgement from the RP regarding the new subtree leading to DR2.

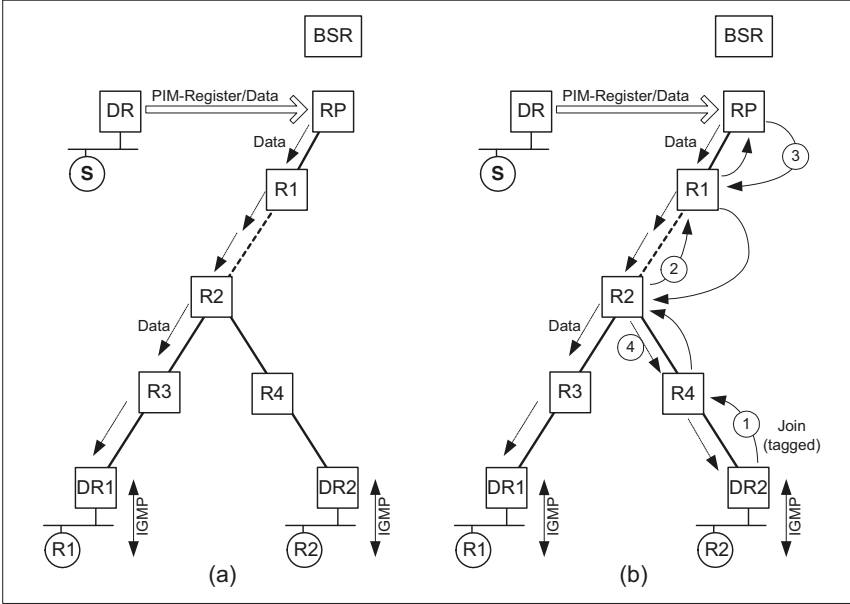


Fig. 4. Receiver Joining Sub-Tree

This is shown in Figure 4 (b), where the Receiver R2 is assumed to have already been authenticated by the DKD and is in possession of the group-key and the unique DR-key (ie.  $K_{dr2}$ ). The Receiver R2 is also assumed to have securely delivered the DR-key ( $K_{dr2}$ ) to its designated router DR2. In Step 1 of Figure 4 (b), the DR2 uses  $K_{dr2}$  to create the tagged-join (as previously proposed in Section 5.2). This tagged-join travels upstream, first to Router R4 and to Router R2. Since Router R2 will become a branching-router for the same distribution tree (ie. same group), it must not as yet deliver data down to Router R4. The Router R2 must first forward the tagged-join towards the RP and wait for an explicit acknowledgement from RP. This is shown as Step 2. In order to be able to recognize the response from the RP, Router R2 must maintain a copy of the tagged-join message obtained from DR2 via Router R4 (more specifically, the nonce and tag of the tagged-join message).

The explicit acknowledgement from the RP takes the form of a *Signed Join-Ack* message sent by the RP towards Router R2 (Step 3). The Signed Join-Ack message consists among others of the elements (*Ack*, *Nonce*, *Tag*, (*Source*, *Group*)), where the *Nonce* and *Tag* are those originally sent within the tagged-join message in Step 1 and Step 2. The acknowledgement message is digitally-signed by the RP using its public key pair.

When Router R2 receives the Signed Join-Ack message from the RP, it will use it to compare against the tagged-join messages which it received earlier (and which it has kept whilst awaiting the explicit acknowledgement from the RP).

A match indicates that the RP has given authorization for the new subtree to be created. That is, in Step 4 of Figure 4 (b), Router R2 allows data to flow through its interface towards Router R4 and finally to DR2. Once data flows, Router R2 can discard or erase both the tagged-join message (issued by DR2 earlier) and the Signed Join-Ack message (from the RP).

If the branching Router R2 does not receive the Signed Join-Ack message from the RP after a given period of time (eg. several heart-beats), it will time-out and discard the tagged-join. In this case data will not flow to the DR2. At this point the behaviour of Router R2 is the same as in the PIM protocol.

Again, similar to other routers, the branching Router R2 does not maintain any host identification information regarding the hosts downstream (in this case, the Receiver R2). This is in conformance with the IP multicast model of [3].

## 6 Remarks and Future Work

In the current work we have analyzed the problem of *receiver access control* in the context of the PIM-SM multicast routing protocol and the multicast distribution tree created by PIM-SM.

The problem of access control for receivers was then explained. Here, one of the possible attacks that exploits the anonymous-receiver underpinnings of IP multicast is one in which a host simply joins a multicast group, without any intention of using the data being delivered to it. In such an attack, the host essentially extends or “pulls” the tree towards the subnet effecting a wastage in resources and state within all the affected routers. In this case, the encryption of the multicast data does not provide any help, since the (encrypted) packets still flows down the distribution tree to the subnet and the (malicious) host.

The solution proposed to counter this deficiency makes use of the fact that the Rendezvous Point (RP) is a good decision-point for access control and that the group-key management event can be extended to facilitate receiver access control.

There are a number of issues that remain to be addressed in the general context of PIM security and in the specific area of member access control:

- *Interdomain sender/receiver access control.* Here, the issue concerns receiver access control when the source and the receiver/DR are located in different PIM domains. The PIM authentication key arrangement of [8] refers only to single PIM domains. The notion of a symmetric key shared by all routers (ie. the equal-opportunity-key) does not scale to multiple PIM domains whose RPs are connected via MSDP [20]. Security issues and possible solutions have been described in [21].
- *The RP as a bottleneck.* Although the RP represents a suitable access control decision point, the advantages from the perspective of security can also become a disadvantage from the perspective of routing performance. Some possible solutions to the bottleneck problem are as follows:
  - Alternative security architectures are deployed, in which the RP (a router) is aided by other network entities (eg. servers) in its task of verifying



join-requests and deciding access control. The notion of other entities – such as servers – aiding routers is not new and can be found in a number of reliable multicast protocols based on the router-assistance concept (eg. GRA [22]) and in some Tree-based ACK scheme (eg. RMTPII [23, 24]). In addition, different key arrangements (over and above the DR-key) can be deployed to create a balance between state held within the routers and the load experienced by the RP.

- A “distributed” access control approach, where the DR-keys are pushed down the multicast distribution tree from the RP and are maintained by the DRs. In this approach, a DR becomes the access control decision point for potential members in its subnet since it will be in possession of the valid DR-keys. Besides requiring the DRs to be trusted, the approach requires the DR-keys to be available at the DRs via transmission through the multicast distribution tree (eg. the special “all multicast routers” group). That is, in itself this approach requires a low-delay and reliable transmission of the DR-keys to the DR. Further research is currently being conducted into this approach.
- *IGMP control message authentication.* Assuming DRs can be trusted, one remaining requirement is that all IGMP messages exchanged between a host and the DR (ie. the multicast router) be source-authentic and integrity-protected. To this end, the DR-key – or more generally an IGMP-key – shared between a host and the multicast router can be used to kickstart a secure channel between the two entities. Again, the GKM event may represent a suitable solution towards distributing the IGMP-key to the host (ie. senders/receivers).
- *The problem of “oscillating switching” attacks in PIM.* The “switch” refers to the basic notion in PIM, where after a given transmission threshold has been reached, a Receiver would switch from the shared-tree (emanating from the RP) to the shortest-path tree emanating from the Source. A possible sophisticated attack in this case consists of the attacker throttling and releasing a critical link in the multicast distribution tree (without severing it) with the aim of effecting a switch back-and-forth by the targeted receivers between the shared-tree and the shortest-path tree. In itself this represents a denial-of-service attack.

These and other issues will be the focus for our future research.

**Acknowledgements:** We would like to thank Brad Cain for his help (and patience) with IGMP and PIM.

## References

1. B. Cain, S. Deering, and A. Thyagarajan, “Internet group management protocol version 3,” Nov 1999. `draft-ietf-idmr-igmp-v3-02.txt` (Work in Progress).
2. H. Holbrook and D. Cheriton, “IP multicast channels: EXPRESS support for large-scale single-source applications,” in *Proceedings of ACM SIGCOMM’99*, (Cambridge, MA), pp. 65–78, ACM, 1999.

3. S. Deering, "Host extensions for IP multicasting," RFC 1112, IETF, 1989.
4. D. Waitzman, C. Partridge, and S. Deering, "Distance vector multicast routing protocol," RFC 1075, IETF, 1988.
5. T. Ballardie, P. Francis, and J. Crowcroft, "Core based trees: An architecture for scalable inter-domain multicast routing," in *Proceedings of ACM SIGCOMM'93*, (San Francisco), pp. 85–95, ACM, 1993.
6. J. Moy, "Multicast extensions to OSPF," RFC 1584, IETF, 1994.
7. S. Deering, D. Estrin, D. Farinacci, M. Handley, A. Helmy, V. Jacobson, C. Liu, P. Sharma, D. Thaler, and L. Wei, "Protocol Independent Multicast – Sparse Mode: Motivations and architecture," Aug 1998. `draft-ietf-pim-arch-05.txt` (Work in Progress).
8. L. Wei, "Authenticating PIM version 2 messages," July 1999. `draft-ietf-pim-v2-auth-00.txt` (Work in Progress).
9. T. Hardjono and B. Cain, "Simple key management protocol for PIM," Mar 1999. `draft-ietf-pim-simplekmp-00.txt` (Work in Progress).
10. S. Kent and R. Atkinson, "IP authentication header," RFC 2402, IETF, Nov 1998.
11. C. Madsen and R. Glenn, "The use of HMAC-MD5-96 within ESP and AH," RFC 2403, IETF, Nov 1998.
12. R. L. Rivest, "The MD5 message digest algorithm," RFC 1321, IETF, Apr 1992.
13. C. Madsen and R. Glenn, "The use of HMAC-SHA-1-96 within ESP and AH," RFC 2404, IETF, Nov 1998.
14. RSA Laboratories, "PKCS1: RSA encryption standard," 1993.
15. T. Hardjono, R. Canetti, M. Baugher, and P. Dinsmore, "Secure IP multicast: Problem areas, framework and building blocks," Nov 1999. `draft-irtf-smug-framework-00.txt` (Work in Progress).
16. T. Hardjono, B. Cain, and N. Doraswamy, "A framework for group key management for multicast security," Feb 1999. `draft-ietf-ipsec-gkmframework-01.txt` (Work in Progress).
17. H. Harney and E. Harder, "Group security association key management protocol," Apr 1999. `draft-harney-sparta-gsakmp-sec-00.txt` (Work in Progress).
18. T. Hardjono, B. Cain, and I. Monga, "Intra-domain group key management protocol," Jul 1999. `draft-ietf-ipsec-intragkm-01.txt` (Work in Progress).
19. C. K. Wong, M. Gouda, and S. Lam, "Secure group communications using key graphs," in *Proceedings of ACM SIGCOMM'98*, ACM, 1998.
20. D. Farinacci, Y. Rekhter, D. Meyer, P. Lothberg, H. Kilmer, and J. Hall, "Multicast Source Discovery Protocol (MSDP)," Jan 2000. `draft-ietf-msdp-spec-03.txt` (Work in Progress).
21. T. Hardjono and B. Cain, "PIM-SM security: Interdomain issues and solutions," in *Communications and Multimedia Security (CMS'99)* (B. Preneel, ed.), (Leuven, Belgium), Kluwer, 1999.
22. B. Cain, T. Speakman, and D. Towsley, "Generic router assist (GRA) building block: Motivation and architecture," Oct 1999. `draft-ietf-rmt-gra-arch-00.txt` (Work in Progress).
23. B. Whetten, M. Basavaiah, S. Paul, and T. Montgomery, "RMTP-II specification," Apr 1998. `draft-whetten-rmtp-ii-00.txt` (Work in Progress).
24. T. Hardjono and B. Whetten, "Security requirements for RMTP-II," Nov 1999. `draft-ietf-rmtp-ii-sec-00.txt` (Work in Progress).

# An Algorithm for Multicast with Multiple QoS Constraints and Dynamic Membership

Aiguo Fei and Mario Gerla

Network Research Lab, Department of Computer Science  
University of California, Los Angeles, CA 90095, USA  
{afei,gerla}@cs.ucla.edu

**Abstract.** In this paper we present an algorithm to construct low-cost source trees for multicast with multiple QoS constraints and dynamic membership. Assuming the availability of link-state information, a join path is computed for a new joining multicast receiver. An algebraic formulation is introduced to show how to determine if the QoS requirements for a new receiver can be satisfied at an intermediate node along the join path and how to adjust the tree without breaking QoS requirements for existing members if they are not. Our scheme builds multicast tree incrementally and thus supports fully dynamic membership. It also supports heterogeneous receivers seamlessly. Moreover, our algorithm can support any number of arbitrary QoS metrics without assuming any dependencies among them, if they satisfy some normal mathematical property. If implemented in a distributed fashion, our approach doesn't require any node to have explicit knowledge of the multicast tree topology, thus it scales well for multicast of large group. Simulation studies have been carried out to study the behavior of our algorithm and compare its performance with other schemes.

## 1 Introduction

In multicast[3], data packets are distributed through a tree structure. A central task in multicasting is to build such a distribution tree (the routing problem). Over the years, many multicast routing algorithms and protocols had been proposed and developed. Several routing protocols had been standardized or are in the process of being standardized by IETF[11, 4] for IP networks, and some of them had been deployed and used on the experimental Mbone and some Internet Service Providers'(ISP) networks[11].

In recent years, great effort has been undertaken to introduce and incorporate quality of service(QoS) into data communication networks such as ATM and IP networks[6]. Many multicast applications, such as video conferencing, multimedia broadcasting, and distance-learning, are QoS-sensitive in nature, thus they will all benefit from the QoS support of the underlying networks. The new challenge is how to build a multicast tree (or multiple trees) to deliver the multicast data from source (or sources) to all receivers so that QoS requirements satisfied and the cost of the multicast tree(s) and/or network resources demanded are

minimized. This problem is NP-complete in general[8] as a constrained Steiner tree problem.

A number of multicast routing algorithms that are QoS-aware have been proposed. Most of these algorithms are “static” algorithms in the sense that they need explicit knowledge of all group members[9,14]. For many such algorithms it is difficult and expensive for members to dynamically join the group since this usually requires a re-computation of the whole tree. Moreover, most of them are not designed to handle heterogeneous receivers (different receivers have different QoS requirements), or multiple QoS constraints. On the other hand, a “dynamic” routing algorithm will build the multicast tree through joining of group members one by one. When a new member joins, the routing algorithm doesn’t reconstruct the whole tree, instead it would try to connect the new receiver to an existing tree member without affecting existing group members. It might be harder for such type of algorithms to produce a globally-optimized multicast tree (in terms of cost), but they are necessary for some applications that have frequent membership dynamics. We anticipate both types of multicast algorithms are needed to support different types of applications.

In this paper, we propose an incremental algorithm to support multicast with dynamic membership and multiple QoS constraints. Assuming the availability of information about the global network (as in an OSPF environment in IP networks), a join route is computed when a new member joins. Our algorithm specifies how this new member would be connected to the existing multicast tree without breaking QoS requirements for existing members and how the existing tree would be adjusted if necessary. The algorithm tries to minimize the total cost of the tree by computing a low-cost join path for new members utilizing a QoS unicast routing algorithm. It supports arbitrary number of QoS metrics, dynamic group membership and heterogeneous receivers. It also has good scalability if implemented in a distributed fashion.

The rest of this paper is organized as follows. Section 2 presents network model and some related work. Sections 3 introduces classification of QoS metrics and definitions necessary for the presentation of the algorithm. Section 4 describes the algorithm, followed by simulation results in section 5. We conclude our paper with a short summary as section 6.

## 2 Network Model and Related Work

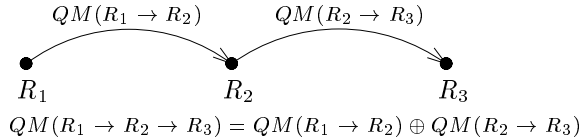
Most existing QoS multicast routing algorithms have real-time applications in mind and have end-to-end delay as the QoS constraint needed to be satisfied. The network is modeled as a connected graph  $G(V, E)$ , where  $V$  is a set of vertices(nodes) and  $E$  is a set of edges(links). For any edge  $e \in E$ ,  $d(e)$  is the delay and  $c(e)$  is the associated cost. One of the node is a traffic source. End-to-end delay on a path from the source to a destination is the summation of  $d(e)$  for all edge  $e$  on the path. The goal is to bound the end-to-end delay on paths from the source to all destinations and minimize the total cost of the tree (as a summation of  $c(e)$  for all  $e \in T$  where  $T$  is the multicast tree). In IP networks,

a source(receiver) node would be a router which has end-user computer in its subnet as a traffic source(receiver) of the multicast group. A survey of those algorithms can be found in [2], and some performance evaluation can be found in [12].

Many centralized multiple-metric(e.g. cost and delay) heuristic algorithms are variations of the well-known single-metric algorithms: Shortest Path Tree(SPT) that optimizes delay from source to all destinations, and Minimum Cost Spanning Tree(MCST) that minimizes the total cost of the tree. The one proposed by V. Kompella et al. in [9] is based on Prim's minimum spanning tree algorithm. In this algorithm, a tree grows from the source, when selecting a new node to join the tree, instead of picking the node connecting to the existing tree via a minimum-cost link, it selects a link that minimizes a given selection function. Another centralized heuristic is Bounded Shortest Multicast Algorithms(BSMA[14]), which starts with an SPT and then replaces some paths in the tree with non-tree paths without breaking the delay constraint but of lower cost. Distributed version of Kompella's algorithm can be found in [10].

As mentioned earlier, most of these algorithms have three main drawbacks: they are not designed to support dynamic membership, they don't consider heterogeneous receivers, and they are difficult to be extended to handle multiple constraints (i.e., jitter and loss in addition to delay). These are the problems our algorithm intends to address.

### 3 QoS Metrics Classification and Definitions



**Fig. 1.** Concatenation of paths.

Let  $R_i$  denote a network node. Consider the concatenation of a path from  $R_1$  to  $R_2$  and a path from  $R_2$  to  $R_3$  to make up a path from  $R_1$  to  $R_3$ . Consider a QoS metric, say,  $QM$ , which is a function of the path. We have  $QM(R_1 \rightarrow R_2)$ ,  $QM(R_2 \rightarrow R_3)$  and  $QM(R_1 \rightarrow R_3)$ , where  $R_i \rightarrow R_j$  denotes a specific path from  $R_i$  to  $R_j$ , and  $R_1 \rightarrow R_3$  refers to the path concatenated by paths  $R_1 \rightarrow R_2$  and  $R_2 \rightarrow R_3$ . In general,  $QM(R_1 \rightarrow R_3)$  is a function of the complete path from  $R_1$  to  $R_3$ . However, for some metric  $QM$ ,  $QM(R_1 \rightarrow R_3)$  only depends on  $QM(R_1 \rightarrow R_2)$  and  $QM(R_2 \rightarrow R_3)$ . For example, delay  $D(R_i \rightarrow R_j)$  is such a metric,

$$D(R_1 \rightarrow R_2 \rightarrow R_3) = D(R_1 \rightarrow R_2) + D(R_2 \rightarrow R_3). \quad (1)$$

To be general, we can define an operator  $\oplus$  on  $QM$  such that,

$$QM(R_1 \rightarrow R_3) = QM(R_1 \rightarrow R_2) \oplus QM(R_2 \rightarrow R_3). \quad (2)$$

Delay is a metric that obeys the regular *addition*(+) operation, we call such a metric as an *additive QoS metric*. Number of hops is another example of additive metrics. Generally delay jitter is also considered to be additive. We call a metric a *transitive QoS metric* if it has the *operator*  $\oplus$  defined as:

$$t_1 \oplus t_2 = \min[t_1, t_2]. \quad (3)$$

or,

$$t_1 \oplus t_2 = \max[t_1, t_2]. \quad (4)$$

To simplify our discussion, from now on, we consider only transitive metric defined by the *min* operator, which is often called *concave* metric in literature. Available bandwidth of a path is a transitive metric, e.g.,  $BW(R_1 \rightarrow R_3) = \min[BW(R_1 \rightarrow R_2), BW(R_2 \rightarrow R_3)]$ .

We call a metric a *multiplicative QoS metric* if it has the *operator*  $\oplus$  defined as:

$$m_1 \oplus m_2 = m_1 \times m_2. \quad (5)$$

If one defines reliability  $r$  as  $r = 1 - \text{loss rate}$ , then  $r$  is a multiplicative metric. On the other hand, if loss rate  $L$  on all link is considered small enough such that  $L(R_1 \rightarrow R_3) = L(R_1 \rightarrow R_2) + L(R_2 \rightarrow R_3)$ , then *loss rate*  $L$  can be considered as an additive metric.

Consider a path from  $S$  to  $D$  that has a *QoS descriptor* denoted as  $QD = \langle a_1, \dots, a_{n_a}, t_1, \dots, t_{n_t}, m_1, \dots, m_{n_m} \rangle$ , where  $a_i$  is an additive QoS metric for the path,  $t_j$  is a transitive metric and  $m_k$  is a multiplicative metric;  $n_a, n_t, n_m$  are number of additive, transitive and multiplicative metrics respectively. To make a meaningful problem, we assume  $a_i \geq 0$  for any additive metric  $a_i$  and  $0 < m_i \leq 1$  for any multiplicative metric  $m_i$ . Consider a *QoS requirement*  $QR$ , which can be denoted in the same way as a *QoS descriptor*,  $QR = \langle a'_1, \dots, a'_{n_a}, t'_1, \dots, t'_{n_t}, m'_1, \dots, m'_{n_m} \rangle$ . We define the  $\leq$  operator between  $QR$  and  $QD$  (and between two  $QD$ 's) as

$$QR \leq QD = \begin{cases} \text{true} & \text{if } a'_i \geq a_i, t'_j \leq t_j \text{ and } m'_k \leq m_k \text{ for all } i, j, k \\ \text{false} & \text{otherwise.} \end{cases} \quad (6)$$

We say a *QoS requirement*  $QR$  is satisfied by the path  $S \rightarrow D$  with *QoS descriptor*  $QD$  if  $QR \leq QD$ .

We can also define the  $\oplus$  operator between two  $QD$ 's as

$$QD \oplus QD' = \langle \dots, x_i \oplus x'_i, \dots \rangle, i \text{ runs over all valid members,} \quad (7)$$

where QoS descriptor  $QD = \langle \dots, x_i, \dots \rangle$  and  $QD' = \langle \dots, x'_i, \dots \rangle$ . For a path  $R_1 \rightarrow R_2 \rightarrow R_3$ ,  $QD(R_1 \rightarrow R_2 \rightarrow R_3) = QD(R_1 \rightarrow R_2) \oplus QD(R_2 \rightarrow R_3)$ .

Consider network nodes  $S$  and  $D$  and an intermediate node  $I$ , the path  $S \rightarrow I$  has QoS descriptor  $QD(S \rightarrow I)$  and the path  $I \rightarrow D$  has QoS descriptor  $QD(I \rightarrow D)$ , we have:

**Lemma 1.** A QoS requirement  $QR$  is satisfied by the path  $S \rightarrow I \rightarrow D$  if  $QR \leq QD(S \rightarrow I) \oplus QD(I \rightarrow D)$ .

To simplify future discussion, if  $QR \leq QD$ , we define a  $\ominus$  operator between them as

$$QR \ominus QD = \langle \dots, a'_i - a_i, \dots, t'_j, \dots, m'_k / m_k, \dots \rangle, \quad (8)$$

where  $QR = \langle \dots, a'_i, \dots, t'_j, \dots, m'_k, \dots \rangle$ , and  $QD = \langle \dots, a_i, \dots, t_j, \dots, m_k, \dots \rangle$ . The  $\ominus$  operator is only defined between a  $QR$  and a  $QD$ , and the result is a new QoS requirement. It is not defined if  $!(QR \leq QD)$ .

These operators defined above obey similar rules as regular arithmetic operators, e.g., if  $QR \ominus QD_2 \leq QD_1$  then  $QR \leq QD_1 \oplus QD_2$ , and vice versa. Thus we have:

**Lemma 2.** Given QoS requirement  $QR$ , a path  $S \rightarrow I$  with  $QD(S \rightarrow I)$  and a path  $I \rightarrow D$  with  $QD(I \rightarrow D)$ ,  $QR$  is satisfied by the path  $S \rightarrow I \rightarrow D$  if  $QR \leq QD(I \rightarrow D)$  and  $QR \ominus QD(I \rightarrow D) \leq QD(S \rightarrow I)$ .

We also introduce a *max* operator on two QoS descriptors as

$$\max[QD, QD'] = \langle \dots, \min[a_i, a'_i], \dots, \max[t_j, t'_j], \dots, \max[m_k, m'_k], \dots \rangle, \quad (9)$$

where QoS descriptor  $QD = \langle \dots, a_i, \dots, t_j, \dots, m_k, \dots \rangle$  and  $QD' = \langle \dots, a'_i, \dots, t'_j, \dots, m'_k, \dots \rangle$ , and  $a_i(a'_i)$  is an additive metric,  $t_j(t'_j)$  is a transit metric and  $m_k(m'_k)$  is a multiplicative metric. Intuitively, if *max* operator is applied on two QoS requirements, the result is a new QoS requirement which is at least as strict as both of the old requirements for any metric. Similarly we can define a *min* operator. By definition, we have

**Lemma 3.** If  $\max[QR_1, QR_2] \leq QD$ , then both  $QR_1 \leq QD$  and  $QR_2 \leq QD$ .

We now make modifications to the multicast tree construction problem in Section 2: instead of delay  $d(e)$ , each edge  $e \in E$  now has QoS characteristics  $QD(e)$ ; for the path (in the multicast tree) from  $s$  to a destination  $t$ , it has to satisfy the QoS requirement s.t.  $QR(t) \leq QD(s \rightarrow t)$  where  $QD(s \rightarrow t) = \sum_{e_i} QD(e_i)$  for all  $e_i$  on the path  $s \rightarrow t$ .

## 4 The Algorithm

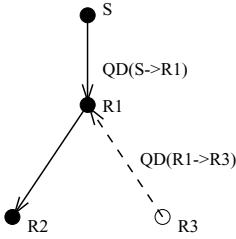
### 4.1 Assumptions and Messages

We assume any QoS metric(requirement) needed to be considered is of one type of those defined above. In this section we describe how our algorithm work in a distributed fashion, the pseudo code for the algorithm can be found in[5]. We also assume the availability of link-state information at each node and a QoS-capable unicast routing algorithm.

Similar to PIM-SM(Protocol Independent Multicast Sparse Mode[4]), we assume a node keeps forwarding information such as source/group, incoming link and set of outgoing link(s). When a new member(receiver) wants to join the group, it sends an explicit **Join request** towards the source. Additionally, in a forwarding entry, each outgoing link is labeled as **active** or **pending**. Multicast traffic is only forwarded to outgoing link(s) marked as active. Two additional types of messages required are **Accept notification** and **Deny notification**. When the Join request is accepted(denied) at some node, an Accept(Deny) notification is sent downstream towards the new receiver. Accept notification will change the corresponding pending flags to active while a Deny notification will clear those pending entries. When a member leaves a group, it sends **Prune message** upstream as in PIM-SM. Additionally, through periodic(or triggered) **QoS Probing messages** from the source (or intermediate nodes), any node in the multicast tree will keep track of QoS characteristics (such as bandwidth reserved, delay, delay jitter and loss, depending on the QoS metrics concerned) on the current path in the multicast tree from the source to that node.

## 4.2 Algorithm Description

When a new receiver(say,  $R_3$ ) wants to join the multicast group with QoS requirement  $QR(R_3)$ , it computes a join path utilizing the available link-state information and a QoS unicast routing algorithm. It then sends a Join message carrying the join path information towards the source.



**Fig. 2.** Join request received by an intermediate node  $R_1$ .

An intermediate node will forward the Join message. It also stamps in the message the QoS characteristics of the link from which the message received, so that when a node(say,  $R_1$ ) receives the Join message from  $R_3$ , the QoS characteristics on the reverse direction of the path it traveled so far is available as  $QD(R_1 \rightarrow R_3)$ . If the node receiving the message is not already in the tree, it creates a new multicast forwarding entry (group/source, in-link, out-link, destination). The out-link is the link from which it receives the message and the in-link is the link to which it should forward the message based on route information carried

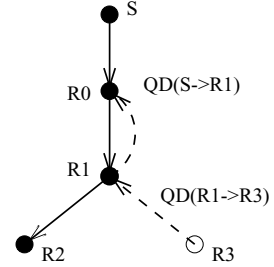
in the message. It then stamps in the Join message the QoS characteristics for the link from which it received the message and forwards that message to the next hop. It also marks the newly created routing entry as “pending” and starts a timer. The routing entry will be marked as active (route pinning) when an Accept message is received. The Accept message is then forwarded downstream. A “pending” routing entry is flushed if its timer goes off.

When a node receives a Join message and it is already part of the multicast distribution tree, say, node  $R_1$  in Fig.2 (where  $S$  is the source), it will check current QoS characteristics  $QD(S \rightarrow R_1)$  and QoS requirements  $QR(R_3)$  and QoS characteristics  $QD(R_1 \rightarrow R_3)$ . If  $QR(R_3) \leq QD(S \rightarrow R_1) \oplus QD(R_1 \rightarrow R_3)$ , by Lemma 1, QoS requirement for new receiver  $R_3$  is satisfied by the path



$S \rightarrow R_1 \rightarrow R_3$ , then the Join request can be accepted at node  $R_1$ . This is as illustrated in Fig.2. In Fig.2 and others to follow, a black node is an in-tree node of the multicast group, non-filled node ( $R_3$ ) is the new joining receiver and a gray node may or may not be an in-tree node. A solid line represents an existing multicast forwarding path and a dashed line represents the path on which the Join request is sent. Multicast packets will be forwarded on the reverse direction of the dashed line if the Join request succeeds. Under the situation illustrated in Fig.2, the Join request is accepted at router  $R_1$ . No additional forwarding is required.

If this is not true, there are two possibilities: (1)the immediate upstream node  $R_0$  of  $R_1$  is the next hop specified by the Join request, (2)the next hop specified in the Join request is not  $R_0$ . In the first case,  $R_1$  can simply add the interface from which it received the Join request to the corresponding multicast routing entry and mark it as pending, and then forward the request to  $R_0$ , which may accept the Join request or forward it further. This is illustrated in Fig.3.



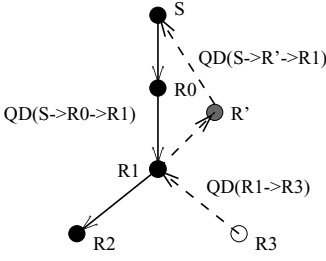
In the latter case, the action to be taken by  $R_1$  is described as follows, which also has two possibilities: (1) $QD(S \rightarrow R_1) \leq QR(R_3) \ominus QD(R_1 \rightarrow R_3)$ , (2)con-

**Fig. 3.**  $!(QR(R_3) \leq QD(S \rightarrow R_1) \oplus QD(R_1 \rightarrow R_3))$ .

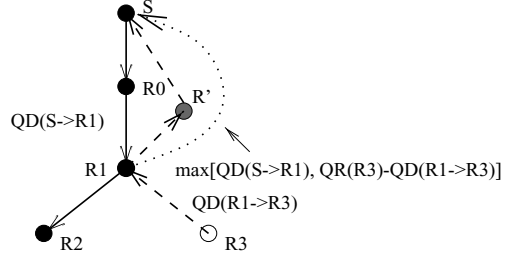
dition (1) is not satisfied. In the first case, let  $R'$  be the next hop specified by the Join request, if the Join request is accepted on the new join path  $S \rightarrow R' \rightarrow R_1$ , then  $QR(R_3) \ominus QD(R_1 \rightarrow R_3) \leq QD(S \rightarrow R' \rightarrow R_1)$ , thus the path  $S \rightarrow R' \rightarrow R_1 \rightarrow R_2$  will also satisfy the QoS requirements for any downstream node  $R_2$  of  $R_1$  by Lemma 2, since we will have  $QR(R_2) \ominus QD(R_1 \rightarrow R_2) \leq QD(S \rightarrow R_1)$  and  $QD(S \rightarrow R_1) \leq QD(S \rightarrow R' \rightarrow R_1)$ . So  $R_1$  will forward Join request to  $R'$ , and mark it as the preferred immediate upstream node. When an Accept message is received later, a QoS probing message is also generated with the current QoS characteristics  $QD(S \rightarrow R_1)$  and multicast to downstream nodes to trigger  $QD$  update. Pruning message is to be sent to its original parent node  $R_0$  to prune itself off from the old branch once multicast traffic comes down from  $R'$ . This is illustrated in Fig.4. In the latter case ( $!(QD(S \rightarrow R_1) \leq QR(R_3) \ominus QD(R_1 \rightarrow R_3))$ ),  $R_1$  can compute a new join path with  $\max[QD(S \rightarrow R_1), QR(R_3) \ominus QD(R_1 \rightarrow R_3)]$  as the QoS requirement. In such a path can be found, QoS requirements for both  $R_3$  and existing downstream nodes of  $R_1$  can be satisfied by Lemma 3 and Lemma 2. Thus  $R_1$  can send a Join request to setup that new path; once it succeeds,  $R_1$  can accept the Join request from  $R_3$  and switch to the new path. If such a path couldn't be found, then  $R_1$  should reject the Join request. This is illustrated in Fig.5.

From the above description, it is easy see that:

**Theorem 1.** If a Join request is accepted, then the new member's QoS requirement is guaranteed to be met without breaking QoS requirement of any existing member.



**Fig. 4.**  $!(QR(R_3) \leq QD(S \rightarrow R_1) \oplus QD(R_1 \rightarrow R_3)), QD(S \rightarrow R_1) \leq QR(R_3) \ominus QD(R_1 \rightarrow R_3)$ .



**Fig. 5.**  $!(QR(R_3) \leq QD(S \rightarrow R_1) \oplus QD(R_1 \rightarrow R_3))$ , and  $!(QD(S \rightarrow R_1) \leq QR(R_3) \ominus QD(R_1 \rightarrow R_3))$ .

### 4.3 QoS Unicast Algorithms

Our algorithm relies on some available QoS unicast routing algorithm to compute join paths. It has been proved that, the shortest-path (or the min-cost path) problem subject to one or more additive or multiplicative constraints is NP-complete[7]. A number of polynomial-time heuristics have been proposed and studied [2]. In our simulation study, we use a simple greedy heuristic based on Bellman-Ford algorithm, more details and pseudo code can be found in [5]. This algorithm can be easily extended to handle multiple constraints and its time complexity is  $O(|V||E|)$ . Of course it does not guarantee a solution even if one exists, due to the NP-completeness nature of the problem.

### 4.4 Complexity Analysis

Let  $n = |V|$  and  $M = |E|$ . At worst case, along the path for a new member to join the existing multicast tree, every intermediate node may need to compute a new join path, so we have:

**Lemma 4.** The computation complexity for a new member to join the group is  $O(hf(n, M))$  at worst case, where  $f(n, M)$  is the complexity of the unicast algorithm applied and  $h$  is the number of hops of the path.

Lemma 4 gives:

**Theorem 2.** The total complexity to setup a multicast group of  $m$  members with our algorithm is  $O(mh_{max}f(n, M))$  at worst case, where  $h_{max}$  is the maximum number of hops of paths and  $h_{max} < n$ .

Algorithms mentioned in section 2 all assumed uniform QoS requirements for all members and only considered delay constraint. To compare our algorithm with them, we have to assume the same conditions:

**Lemma 5.** If there is only one uniform single additive or multiplicative metric constraint (in addition to a cost-minimization objective), the computation complexity for a new member to join the group is  $O(f(n, M) + n)$ , in a centralized implementation and in a distributed implementation assuming accurate link-state information (e.g., no stale information).

**Proof:** See [5].

Following Lemma 5, we have:

**Theorem 3.** When there is a single constraint of an additive or multiplicative metric, the total complexity to setup a multicast group of  $m$  members with our algorithm is  $O(mf(n, M) + mn)$ .

Using a constrained unicast routing algorithm with complexity  $f(n, M) = O(nM)$  as mentioned earlier, our algorithm could build a multicast tree of  $m$  members with complexity  $O(mnM)$ . In most realistic networks,  $M = O(n)$ , thus the complexity is  $O(mn^2)$ . Table 1 gives a complexity comparison with several other algorithms:

**Table 1.** Comparison of computation complexity to build a multicast tree, using a  $O(nM)$  unicast algorithm and assuming  $|E| = O(|V|)$ . For centralized algorithms, it assumed that all group members are known in advance and the algorithms build trees for all members at once.

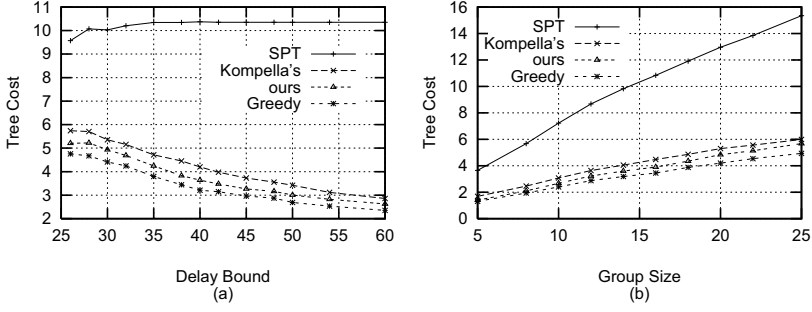
algorithm	complexity
our algorithm	$O(mn^2)$
SPT (centralized)	$O(n \log n)$
SPT (distributed)	$O(mn \log n)$
Kompella's(centralized)	$O(n^3)$
BSMA(centralized)	$O(n^3 \log n)$

## 5 Simulation Results

Simulation has been used to study the behavior of our algorithm and to compare its performance with other schemes. In our first set of simulations, uniform bandwidth and delay constraints are considered, so we can compare our scheme with others. Three other algorithms are simulated: SPT(Shortest Path Tree) algorithm (single metric on delay), Kompella's algorithm([9]), and a greedy algorithm[1]. In the greedy algorithm, each time a node joins the group, it computes a delay-bounded min-cost path to all members in the existing tree. Then it chooses the one with lowest cost. The unicast QoS routing algorithm used for our algorithm is a delay-bounded min-cost heuristic algorithm based on Bellman-Ford algorithm[5].

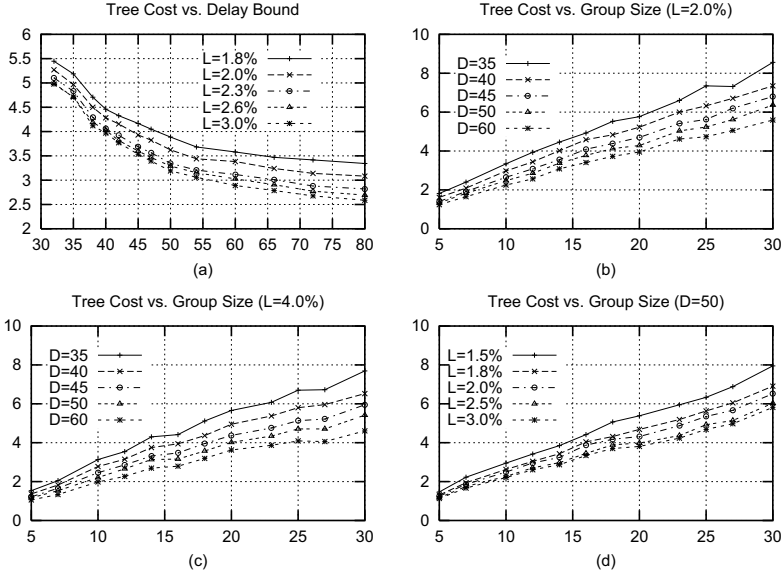
We use randomly generated networks using the approach given in [13]. In simulations presented here, networks have a fixed size of 60 nodes chosen over a  $30 \times 30$  grid. Parameters are chosen such that in average each node has a degree of 4 or 5. Geometric distance is used as delay on a link. To be as general as possible, a random cost between 0 to 1 is generated for each link. For simplicity, links are assumed to be bi-directional and symmetric. Furthermore, all links are assumed to have enough bandwidth, so bandwidth is not explicitly considered in the simulation.

Two sets of results for the comparison are shown here in Figure 6 (a) and (b). Each point in the figures represent the average from 100 instances. For each instance, we randomly pick a node as source node and a given number of



**Fig. 6.** (a) Tree cost vs. delay bound with fixed group size (=15); (b) Tree cost vs. group size with fixed delay bound(=40).

other nodes as receivers. In SPT and Kompella's algorithm, a tree is built for all members. In our algorithm and greedy algorithm, nodes join the group one by one in a random order. Fig.6(a) shows the tree cost vs. delay bound with fixed group size 15. Fig.6(b) shows the results for experiments with a fixed delay bound 40 and group size ranging from 5 to 25.



**Fig. 7.** Tree cost with varied delay bounds (D) and loss probability bounds (L).

As one can see, SPT always builds trees of higher cost than all others and almost don't change any with different delay. This is not surprising since SPT only builds a tree based on a single delay metric and makes no attempt to

optimize the tree in terms of cost. Of the remaining schemes, greedy algorithm always produce the lowest-cost tree. This is expected since it requires much more computation than others. Our algorithm works between Kompella's and greedy algorithms. Fig.6(a) shows all the three algorithms indeed can reduce the tree cost when delay bound is relaxed. In Fig.6(b), when group size grows, the cost of trees produced by all three algorithms increases at a rate lower than SPT. In summary, Kompella's, greedy and our algorithm produce trees of similar costs. However, compared with Kompella's algorithm, ours has the advantage of being fully distributed and allowing incremental tree build-up to accommodate dynamic joining members. Though greedy algorithm has the advantage of slightly lower cost, it is much more costly in terms of computation overhead.

To show our algorithm is effective when there are more than one constraint, we generate networks with one more random number (from 0 to 1%) associated with each link as loss probability. Such random loss can be considered to be caused by various reasons (transmission error, lossy wireless links, etc.). Because of the real-time nature of the applications, retransmission is not feasible and so we want to bound the total loss probability. Fig.7(a) shows tree cost vs. delay bound, similar to Fig.6(a), but now different curves are for different loss probability bounds. All curves have the same trend of decreasing with the relaxation of delay bound, as in Fig.6(a). It also shows that when loss probability bound is relaxed, the tree cost decreases along with. Fig.7(b), (c) and (d) present tree cost vs. group size with varied delay bounds and loss probability bounds. Fig.7(b) shows curves for different delay bounds with a fixed loss probability bound, Fig.7(c) shows curves of the same group of delay bounds but with a different fixed loss probability bound. Both demonstrate lower tree cost for more relaxed delay bound. Fig.7(c) has a more relaxed loss probability bound than (b), so one can see that each tree cost in (c) is lower than that in (b) with same group size and delay bound. Fig.7(d) shows a similar group of curves with a fixed delay bound but varied loss probability bounds. All these show that our algorithm is effective in lowering tree cost when it is allowed by the constraints.

## 6 Conclusions

In this paper, we have presented an incremental multicast tree construction algorithm. Assuming network information available at each node, it is possible to incrementally grow the tree through dynamic joining of group members. Our algorithm builds low-cost tree by utilizing unicast algorithm to compute a minimum-cost join path. Our algorithm specifies how to adjust the tree when necessary to accommodate the QoS constraints of new members without breaking QoS of existing members, thus enhances the possibility of new member being accepted without recomputing the whole tree. In addition to its distributed nature and dynamic membership support, our mechanism can support heterogeneous receivers with any number of QoS metrics without any inter-metric dependency assumptions. Moreover, since a node doesn't need any explicit knowledge of existing receivers when it computes a join path, there is no requirement for

any node to maintain multicast tree topology information. Thus our approach can scale well to support multicast with large number of participants. Simulation results for delay-constrained multicast cast show that it has performance comparable to that of other schemes in terms of tree cost but with lower computation complexity. Additional simulation shows it performs consistently with more constraints. The key contribution of our work is an algorithm capable of handling multiple QoS constraints and supporting heterogeneous receivers with dynamic memberships.

## References

1. D. Cavendish, A. Fei, M. Gerla, and R. Rom. On the maintenance of low cost multicast trees with bandwidth reservation. In *Internet Mini-Conference with Globcom98*, Australia, 1998.
2. S. Chen, and K. Klara. An overview of quality of service routing for next-generation high-speed networks: problems and solutions. In *IEEE Network Magazine*, pp.64-79, November/December 1998.
3. C. Diot, W. Dabbous, and J. Crowcroft. Multipoint communication: a survey of protocols, functions, and mechanisms. In *IEEE Journal on Selected Areas in Communications*, Vol.15(3), pp.277-290, April 1997.
4. S. Deering, D. Estrin, D. Farinacci, et al.. Protocol independent multicast-sparse mode (PIM-SM): motivation and architecture. Internet draft: draft-ietf-idmr-pim-arch-05.txt{ps}, August 1998.
5. A. Fei, and M. Gerla. Receiver-initiated multicasting with multiple QoS constraints. *Technical Report No.990043*, Department of Computer Science, UCLA, 1999.
6. P. Ferguson and G. Huston. *Quality of Service*, John Wiley & Sons, Inc., 1998.
7. M. Garey, and D. Johnson. *Computers and Intractability, a Guide to the Theory of NP-Completeness*, W.H. Freeman and Company, New York, 1979.
8. F. Hwang, D. Richards, and P. Winter. *The Steiner Tree Problem*, North-Holland, 1992.
9. V. P. Kompella, J. C. Pasquale, and G. C. Polyzos. Multicast routing for multimedia communication. In *IEEE/ACM Transactions on Networking*, Vol.1(3), pp.286-292, June 1993.
10. V. P. Kompella, J. C. Pasquale, and G. C. Polyzos. Two distributed algorithms for the constrained Steiner tree problem. In *Proceedings of Computer Communication Networking*, June 1993.
11. D. Kosiur. *IP Multicasting*. John Wiley & Sons, Inc., 1998.
12. H. Salama, D. S. Reeves, and Y. Viniotis. Evaluation of multicast routing algorithms for real-time communications on high-speed networks. In *IEEE Journal of Selected Areas in Communications*, Vol.15(3), pp.332-344, April 1997.
13. B. M. Waxman. Routing of multipoint connections. In *IEEE Journal of Selected Areas in Communications*, Vol.6(9), pp.1617-1622, December 1988.
14. Q. Zhu, M. Parsa, and J. Garcia-Luna-Aceves. A source-based algorithm for delay-constrained minimum-cost multicasting. In *Proceedings of IEEE Infocom'95*, pp.377-385, 1995.

# Optimal Traffic Partitioning in MPLS Networks

Esmael Dinan<sup>1</sup>, Daniel O. Awduche<sup>2</sup>, and Bijan Jabbari<sup>1</sup>

<sup>1</sup> George Mason University, Fairfax, VA 22030, USA  
edinan1@gmu.edu, bjabbari@gmu.edu

<sup>2</sup> UUNET (MCI Worldcom), 22001 Loudoun County Parkway,  
Ashburn, VA 20147, USA, awduche@uu.net

**Abstract.** Multiprotocol Label Switching (MPLS) is an emerging Internet technology that facilitates traffic engineering in service provider networks. This paper considers a network performance optimization problem related to traffic engineering over MPLS. We investigate the issue of dynamic partitioning of MPLS ingress traffic into several parallel Label Switched Paths (LSP). Specifically, we present a stochastic framework for the traffic partitioning problem. Within this framework, a set of parallel edge disjoint LSPs is modeled by parallel queues and a partitioning algorithm is devised for different service classes that is adaptive to the prevailing state of the network. The performance of this approach is illustrated by numerical examples.

## 1 Introduction

As the Internet continues to grow exponentially and becomes more mission critical, the need for performance optimization of service provider networks through effective and efficient resource management becomes imperative. Performance optimization of operational networks is achieved through traffic engineering. The basic problem of traffic engineering in service provider networks concerns the mapping of traffic onto the network infrastructure, such that traffic oriented and resource oriented performance objectives are achieved. Traffic oriented performance characteristics are usually expressed in terms of QoS parameters, such as loss, delay, and goodput. On the other hand, resource oriented performance objectives relate to the efficient utilization of network assets.

Multiprotocol Label Switching (MPLS) is one of the most promising technologies for traffic engineering in IP networks [2]. MPLS is an approach to packet forwarding whereby short fixed length labels are attached to packets by a label switching router (LSR) at the ingress to an MPLS domain [1]. An ingress LSR attaches labels to packets based on the concept of forwarding equivalence classes (FEC), so that packets that belong to the same FEC are assigned the same label value. As labeled packets traverse the MPLS domain, packet forwarding is performed according to the classical label swapping paradigm. MPLS becomes a powerful abstraction for traffic engineering when coupled with a signaling protocol that supports explicit LSP setup capability [3].

The requirements for traffic engineering over MPLS are outlined in [2]. These requirements include a set of capabilities that facilitate the realization of a variety of network optimization policies. Regardless of the specific optimization policy employed in a given network, three fundamental problems related to traffic engineering over MPLS were identified in [2]. The first problem concerns how to partition traffic into traffic trunks or forwarding equivalence classes. The second problem concerns how to map traffic trunks onto label switched paths (LSPs). The third problem concerns how to map traffic trunks onto the physical network topology through the selection of routes for LSPs.

This paper describes an analytical approach to network performance optimization which relates to the first two aspects of the MPLS traffic engineering problem. Specifically, we present a stochastic framework for dynamic traffic partitioning by which more efficient use of alternate paths can be made in MPLS domains. We suppose that a set of parallel edge disjoint LSPs have been pre-established between an ingress node and an egress node. We describe a procedure for partitioning ingress traffic into forwarding equivalence classes, and a traffic assignment algorithm for mapping the FECs onto the pre-established parallel LSPs, taking into account the dynamics of network state.

The traffic partitioning problem as presented here is peculiar to MPLS and is distinguished from ATM based traffic management mechanisms. A specific approach to MPLS adaptive traffic engineering (called MATE) based on a partitioning paradigm was presented in [8]. The MATE proposal maps input traffic to different LSPs based on system parameters measured by probe packets.

This paper is organized as follows: Section 2 describes the networking context and associated system assumptions. Section 3 presents the analytical model, algorithms, and performance analysis. Transient behavior of the system is studied in section 4. Section 5 illustrates the concepts through numerical examples. Finally, section 6 provides concluding remarks.

## 2 The Traffic Partitioning Problem in MPLS Networks

We consider an MPLS network in which the input traffic to an ingress label switching router (LSR) is dynamically partitioned and mapped onto several label switched paths. Figure 1 depicts an example of such a network. Given this network, the problems that we address in this paper are two fold. The first aspect concerns the partitioning of ingress traffic into FECs. The second aspect concerns the mapping of FECs onto pre-established LSPs. The intent is to perform the partitioning and mapping operations in such a way that traffic oriented performance characteristics are enhanced, while network resources are utilized efficiently. Since the characteristics of ingress traffic, the membership of FECs, and state of the network vary with time, the optimal partitioning procedure needs to be dynamic and adaptive.

In a differentiated services Internet, the ingress traffic will consist of a collection of traffic aggregates belonging to different service classes with different forwarding treatment requirements which are specified in terms of per-hop for-



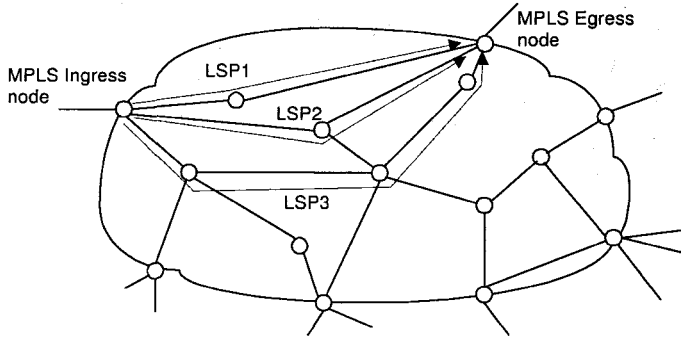


Fig. 1. An example MPLS network.

warding behaviors (PHB) encoded in the packet headers (see e.g., [4]). In the following, the performance measures for transmission quality will be the probability of packet loss and packet transmission delay. The traffic mapping is done in two stages. First, conceptually, the input traffic is partitioned into  $N$  bins at the MPLS ingress node, each with approximately  $r = \lambda/N$  packets/sec. Then the traffic from the  $N$  bins is mapped onto  $n$  pre-established LSPs. With this method, the partitioning ratio is a discrete variable,  $i/N$ ,  $1 \leq i \leq N$ . A simple method to partition the input traffic is on a per-packet basis, for example in a round-robin fashion. However, this method suffers from the possibility of excessive packet reordering and is not recommended in practice. Another well known method is to filter the ingress using a hash function on a subset of the IP packet header (see e.g., [5]).

### 3 System Model and Analysis

We focus on a single MPLS ingress LSR which is connected to an egress LSR via  $n$  parallel LSPs. Let's denote such set of LSPs by  $\Psi = \{\varphi_i : i = 1, \dots, n\}$ . We model each hop through which an LSP traverses by a queue (specifically an M/M/1/K queue), so that each LSP is effectively represented by a sequence of such queues. As shown in Figure 2, the system essentially consists of a network of queues. Let  $Q = \{Q_{ij}; i = 1, \dots, n, j = 1, \dots, R_i\}$  be the set of queues comprising the queueing network, where  $R_i$  is the number of hops traversed by  $\varphi_i \in \Psi$ . Let  $\lambda(t)$  be the time-dependent aggregate arrival rate at the ingress LSR of the parallel LSPs destined for the egress LSR. We assume that the traffic arrival rate at each nodal queue traversed over LSP  $\varphi_i \in \Psi$  is a function of time and can be divided into two parts: (1) the flow rate  $\lambda_i(t) = p_i \lambda(t)$ , which is the time-dependent contribution of  $\varphi_i$  and takes into account the traffic mapping probability vector  $P = \{p_i : i = 1, \dots, n; \sum_{i=1}^n p_i = 1\}$ ;  $\lambda_{ij}(t)$  represents this flow passing through  $Q_{ij}$  and (2) the variable  $\gamma_{ij}(t)$ , which represents the time-dependent contribution from all other LSPs that traverse the node. Let  $\mu_{ij}$  be

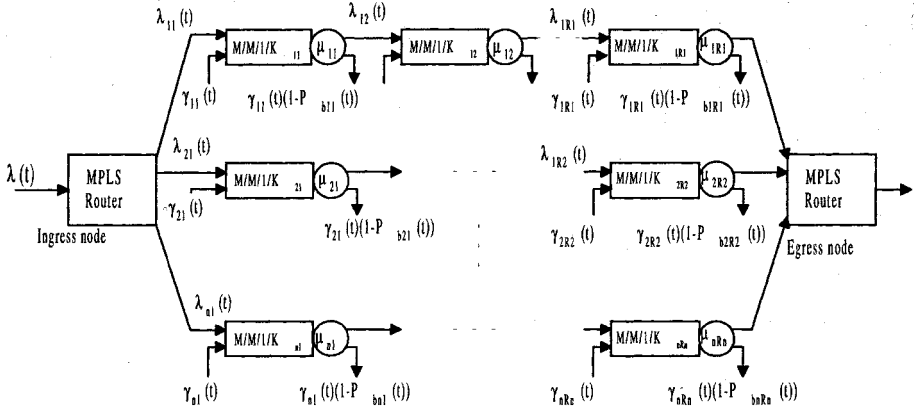


Fig. 2. System model considering a series of queues for each LSP.

the service rate associated with queue  $j$  on LSP  $\varphi_i$ . The available bandwidth at each queue is the difference between the average service rate and the average aggregate arrival rate of all traffic that traverse the queue. The methods for measuring the available bandwidth of a given path have been described in [6]. We suppose that the variation of  $\lambda(t)$  and  $\gamma_{ij}(t)$ ,  $1 \leq i \leq n$ ,  $1 \leq j \leq R_i$ , are slow enough with respect to time, so that we can consider only the steady-state behavior of the queues in the analysis.

### 3.1 Best Effort Traffic

First, we consider only best effort traffic. We define the vector

$\Lambda(t) = [\lambda_1(t), \lambda_2(t), \dots, \lambda_n(t)]^T$  as the input traffic rates to LSPs  $\varphi_i \in \Psi$ , and  $\Lambda_i(t) = [\lambda_{ti1}(t), \lambda_{ti2}(t), \dots, \lambda_{tiR_i}(t)]^T = \Lambda_i + \Gamma_i$  where  $\lambda_{tij}(t) = \lambda_{ij}(t) + \gamma_{ij}(t)$ , as the vector of queue arrival rates and  $M_i = [\mu_{i1}, \mu_{i2}, \dots, \mu_{iR_i}]^T$  as the set of queue service rates. The system performance for best effort is measured by the total number of lost packets. The objective of the analysis is to find the traffic mapping vector  $P(t) = [p_1(t), p_2(t), \dots, p_n(t)]^T$  that optimizes the system performance. We have

$$[\lambda_1(t), \lambda_2(t), \dots, \lambda_n(t)]^T = [p_1(t), p_2(t), \dots, p_n(t)]^T \lambda(t) \quad (1)$$

and

$$\sum_{i=1}^n p_i(t) = 1 \quad (2)$$

We need to find the column vector  $P(t)$  such that the loss rate

$$L(t) = P_{b1}(t)\lambda_1(t) + P_{b2}(t)\lambda_2(t) + \dots + P_{bn}(t)\lambda_n(t) = \sum_{i=1}^n P_{bi}(t)\lambda_i(t) = P_b(t)^T \Lambda(t) \quad (3)$$

is minimized, where  $P_b(t) = [P_{b1}(t), P_{b2}(t), \dots, P_{bn}(t)]^T$ . The parameter  $P_{bi}$  is the average probability of loss for the traffic of LSP  $\varphi_i$ ,  $\Lambda_i$ . We define  $P_{bij}$  as the probability of loss in queue  $Q_{ij}$  with the arrival rate  $\lambda_{tij}(t)$ . The loss rate of LSP  $\varphi_i$  traffic in this queue is equal to

$$L_{ij}(t) = P_{bij}(\lambda_{tij}(t), \mu_{ij})\lambda_{ij}(t) \\ = P_{bij}(\lambda_{tij}(t), \mu_{ij})\lambda_{i(j-1)}(t) \left(1 - P_{bi(j-1)}(\lambda_{ti(j-1)}(t), \mu_{i(j-1)})\right) \quad (4)$$

where  $\lambda_{ij}(t) = \lambda_{i(j-1)}(t) \left(1 - P_{bi(j-1)}(\lambda_{ti(j-1)}(t), \mu_{i(j-1)})\right)$  and  $\lambda_{i1}(t) = p_1(t)\lambda(t)$ .

Assuming blocking to be relatively small in the network ( $P_{bij} \ll 1$ ), the departure of a queue with Poisson arrival rate can be approximated by another Poisson process. Therefore, each queue in the system can be approximated by an equivalent M/M/1/K queue. The total loss rate,  $L(t)$ , can be rewritten as follows

$$L(t) = \sum_{i=1}^n \left( \sum_{j=1}^{R_i} P_{bij}(\lambda_{tij}(t), \mu_{ij})\lambda_{ij}(t) \right) \quad (5)$$

where for a fixed time  $t_0$ ,  $K_{ij}, \mu_{ij}$  and  $\gamma_{ij}(t_0)$ ,  $1 \leq i \leq n$  and  $1 \leq j \leq R_i$ , are known. The variables  $\lambda_i(t)$ ,  $1 \leq i \leq n$ , are to be determined such that  $\lambda(t) = \sum_{i=1}^n \lambda_i(t)$ , and  $L(t)$  is minimized. The set  $\{\lambda_i(t), 1 \leq i \leq n\}$  that minimizes  $L(t)$  can be found by iteration or Lagrange Multiplier methods [7].

Since the queue arrival rate vectors  $\Gamma_i(t)$ , and the Ingress node arrival rate  $\lambda(t)$  are functions of time, an iterative algorithm needs to be executed in every time interval  $\Delta t$ , hence a significant computational effort is involved. Therefore, we present a simple method to calculate the optimal vector  $p(t + \Delta t)$  when the value of the vector  $p(t)$  is known at time  $t$ . With this technique, the minimum value of  $L(t)$  needs to be obtained just one time, and then  $p(t)$  is updated periodically with a simple algorithm in each time step  $\Delta t$ .

$L_i(\lambda_i(t), M_i)$  represents the loss rate in the path  $\varphi_i$  and is equal to

$$L_i(\lambda_i(t), M_i) = P_{bi}(t)\lambda_i(t) = \sum_{j=1}^{R_i} P_{bij}(\lambda_{tij}(t), \mu_{ij})\lambda_{ij}(t) \quad (6)$$

We define function  $S_{\Gamma_i}(t)$  as the sensitivity of the loss rate to the other paths traffic rate, and the function  $S_{\lambda_i}(t)$  as the sensitivity of the loss rate to the path arrival rate, where the following holds

$$: S_{\Gamma_i}(t) = \left[ \frac{\partial L_i(\lambda_i(t), M_i)}{\partial \gamma_{i1}(t)}, \frac{\partial L_i(\lambda_i(t), M_i)}{\partial \gamma_{i2}(t)}, \dots, \frac{\partial L_i(\lambda_i(t), M_i)}{\partial \gamma_{iR_i}(t)} \right] \quad (7)$$

$$S_{\lambda_i}(t) = \frac{\partial L_i(\lambda_i(t), M_i)}{\partial \lambda_i(t)} \quad (8)$$

In a simplified model where each LSP is modeled by just one M/M/1/K queue (see Figure ??), which presents the bottleneck of the path, we have:

$$\begin{aligned} S_{\lambda_{pi}}(t) &= S_{\lambda_i}(t) = \frac{\partial L_i(\lambda_i(t), \mu_i)}{\partial \lambda_i(t)} \\ &= \rho_i(t)^{K_i} \frac{(1 + K_i) (1 - \rho_i(t)^{K_i+2}) - (2 + K_i) (\rho_i(t) - \rho_i(t)^{K_i+2})}{(1 - \rho_i(t)^{K_i+1})^2} \end{aligned} \quad (9)$$

Since the loss rate increases with increasing arrival rate, the elements of  $S_{\Gamma_i}(t)$  and  $S_{\lambda_i}(t)$  are positive.

We now consider two different cases. First, we consider the effect of the variation of the vectors  $\Gamma_i(t)$  on the mapping vector,  $P(t)$ . Then, we consider the effect of the variation in the Ingress node arrival traffic rate  $\lambda(t)$ . In the simplest case, we suppose that during time  $\Delta t$  only the arrival rate of queues in LSP  $\varphi_i$  changes, and we have

$$\Gamma_i(t + \Delta t) = \Gamma_i(t) + \Delta \Gamma_i(t), \quad \Delta \Gamma_i(t) = [\Delta \gamma_{i1}(t), \Delta \gamma_{i2}(t), \dots, \Delta \gamma_{iR_i}(t)]$$

$$\Gamma_k(t + \Delta t) = \Gamma_k(t), \quad 1 \leq k \leq n, k \neq i \quad (10)$$

To compensate for the effect of variation of the arrival rates on the  $L_i(\cdot, \cdot)$ , we need to change the arrival rate of the LSP  $\varphi_i$  in a way that loss rate remains unchanged. This implies that:

$$\begin{aligned} L_i(\lambda_i(t + \Delta t), M_i) &= L_i(\lambda_i(t), M_i) + \sum_{j=1}^{R_i} S_{\Gamma_{ij}}(t) \Delta \gamma_{ij}(t) + S_{\lambda_i}(t) \Delta \lambda_i \\ &= L_i(\lambda_i(t), M_i) \end{aligned} \quad (11)$$

where  $\lambda_i(t + \Delta t) = \lambda_i(t) + \Delta \lambda_i$ , and linear approximation has been used for the loss rate changes during time interval  $\Delta t$ . Then,

$$\Delta \lambda_i = - \frac{\sum_{j=1}^{R_i} S_{\Gamma_{ij}}(t) \Delta \gamma_{ij}(t)}{S_{\lambda_i}(t)} \quad (12)$$

Therefore, to compensate for the effect of  $\Delta \Gamma_i(t)$  of the queues of LSP<sub>*i*</sub>, we change the arrival rate of LSP  $\varphi_i$  by  $\Delta \lambda_i$  (that may be positive or negative). Now, we need to find a method to distribute the extra traffic  $-\Delta \lambda_i$  (let's say *delta traffic*) on different paths. We present the following *Traffic Assignment Algorithm* which can be used to map the traffic  $(-\Delta \lambda_i)$  to a path such that  $L(t)$  remains minimum:

*Traffic Assignment Algorithm for the delta traffic ( $-\Delta\lambda_i$ ):*

if ( $-\Delta\lambda_i > 0$ )

Shift the traffic ( $-\Delta\lambda_i$ ) from path  $\varphi_i$  to the path with the lowest value of  $S_\lambda(t)$ .

else ( $(-\Delta\lambda_i) < 0$ )

Shift the traffic ( $\Delta\lambda_i$ ) from the path with the highest value of  $S_\lambda(t)$  to path  $\varphi_i$ .

end

The main idea is that if ( $-\Delta\lambda_i$ )  $> 0$ , we add the delta traffic to the path whose loss rate is the least sensitive to  $\lambda$ , and therefore we minimize the increase of  $P_b$ . If ( $-\Delta\lambda_i$ )  $< 0$ , we decrease the traffic of the path whose loss rate is the most sensitive to  $\lambda$ , and therefore we maximize the decrease of  $P_b$ .

Now, consider a more complex case, wherein not only the arrival rate of all queues but also the arrival rate to the ingress node change during time interval  $\Delta t$ , i.e.,  $\Gamma_i(t + \Delta t) = \Gamma_i(t) + \Delta\Gamma_i(t)$ , where  $1 \leq i \leq n$ , and  $\lambda(t + \Delta t) = \lambda(t) + \Delta\lambda(t)$ .

The following algorithm re-assigns the input arrival traffic to different queues:

1. Calculate  $\Delta\lambda_i = \frac{\sum_{j=1}^{R_i} S_{\Gamma_{ij}} \Delta\lambda_{ij}}{S_{\lambda_i}(t)}$  for  $1 \leq i \leq n$ .

2. Assign the  $n+1$  delta arrival traffic ( $\Delta\lambda$  and  $\Delta\lambda_i$ ,  $1 \leq i \leq n$ ) to the queues as follows:

For each delta arrival traffic:

a. Execute the *Traffic Assignment Algorithm*.

b. Update variables  $P(t)$  and  $\Lambda(t)$ .

The above algorithm assigns the traffic to the queues in a way to keep the loss rate minimum. The closeness of the partitioning vector obtained by the above algorithms and the optimum one depends on the error of the linear approximation used in Eq. ???. Smaller changes during time interval  $\Delta t$  lead to more exact results. If the number of traffic bins shifted in each run of the algorithm is higher than a certain value ( $\Delta\lambda_{\max}$ ), the time interval  $\Delta t$  can be decreased in order to obtain a better approximation. The parameter  $\Delta t$  can be controlled dynamically based on the system parameter variation in order to obtain exact results. It is notable that (as described in section II), the elements of the vector  $P(t)$  can only have discrete values,  $i/N$ ,  $1 \leq i \leq N$ , where  $N$  is the number of bins that the input traffic is mapped. Therefore, when  $\Delta\lambda$  is calculated, it should be rounded to a discrete number of bins.

### 3.2 TCP Traffic

Another interesting case is when both delay and loss are important factors in system performance optimization. A good example is TCP traffic, which is an important type of traffic in contemporary data networks. Since TCP reacts to packet losses, the total system goodput and packet delay are affected by both delay and loss. It is well-known that TCP achieves bandwidth that is inversely proportional to the square root of the packet loss probability  $P_b(t)$ , under idealized conditions. Note that this approximation is reasonable in a certain range

of  $P_b$  (not too close to zero or one). Since the bandwidth achieved by TCP is reduced by a factor of square root of  $P_b$ , the total delay will be proportional to  $\sqrt{P_b}\tau$ . In this case the optimal mapping vector should be calculated to minimize

$$\tau_{avg} = \sqrt{P_{b1}(t)}\tau_1(t)P_1(t) + \dots + \sqrt{P_{bn}(t)}\tau_n(t)P_n(t) = \sum_{i=1}^n \sqrt{P_{bi}(t)}\tau_i(t)P_i(t) \quad (13)$$

where

$$\tau_{tcp,i}(\lambda_i(t), \mu_i) = \sqrt{P_{bi}(\lambda_i(t), \mu_i)}\tau_i(\lambda_i(t), \mu_i)P_i(t) \quad (14)$$

And the sensitivity functions will be

$$S_{\Gamma_i}(t) = \left[ \frac{\partial \tau_{tcp,i}(\lambda_i(t), M_i)}{\partial \gamma_{i1}(t)}, \frac{\partial \tau_{tcp,i}(\lambda_i(t), M_i)}{\partial \gamma_{i2}(t)}, \dots, \frac{\partial \tau_{tcp,i}(\lambda_i(t), M_i)}{\partial \gamma_{iR_i}(t)} \right] \quad (15)$$

$$S_{\lambda_i}(t) = \frac{\partial \tau_{tcp,i}(\lambda_i(t), M_i(t))}{\partial \lambda_i(t)} \quad (16)$$

The algorithm described previously can be used for TCP traffic partitioning with the above parameters.

## 4 Transient Behavior and System Stability

Now let us consider the transient response of the system when a link is shared by more than one LSP (as an example see Figure 5). Traffic assignment algorithm is executed in all Ingress nodes of the MPLS network. The MPLS Ingress nodes dynamically partition the input traffic based on the current state of the network. Furthermore, there is no need for time synchronization among Ingress nodes in the algorithm. Since a link might be shared by more than one label switched path, a change of traffic partitioning table in one Ingress node can be interpreted as a change of network state by other Ingress nodes. After  $\Delta t$ , these Ingress nodes will execute the traffic assignment algorithm and change again their traffic partitioning table. At a stable equilibrium point any small excursion of the system state variables is forced back to another equilibrium values. Otherwise, for an unstable equilibrium point, a perturbation of the state variables is forced further away from an equilibrium point. The model presented in Figure 2 and the developed analysis let us to follow the transient and steady state behavior of different architectures due to any input traffic or system state changes.

Let  $N$  be the number of Ingress nodes in the system, and  $\Lambda_I = [\lambda_{I1}, \dots, \lambda_{IN}]$  the input rate to the Ingress nodes. We suppose that during time  $\Delta t$  the arrival rate of Ingress nodes changes by  $\Delta \Lambda_{In} = [\Delta \lambda_1, \dots, \Delta \lambda_2]$ . As described in section III, the arrival rate of each nodal queue in LSP  $i$  of Ingress node  $k$  is a function of time and can be divided into two parts: traffic of LSP  $i$ ,  $(\lambda_{ij})_k$ , which is a part of the arrival rate traffic  $\lambda_{Ik}$ , and traffic of other paths input to the node,  $(\gamma_{ij})_k$ .

Other paths traffic (or load traffic) represents the time dependent traffic load of the node. To calculate the *load traffic* of each link, we suppose that packet loss is negligible. Therefore, we will have the following relation for the load traffic,  $(\gamma_{ij})_k$ , which represents load traffic of link  $j$  of LSP <sub>$i$</sub>  of Ingress node  $k$ .

$$(\gamma_{ij})_k = \sum_{k=1}^{N_I} \sum_{n=1}^{LSP_k} a_{kn} (\lambda_{In})_k \quad (17)$$

where  $a_{kn} = \begin{cases} 1, & \text{If the link is shared by LSP } n \text{ of ingress node } k \\ 0, & \text{If the link is not shared by LSP } n \text{ of Ingress node } k \end{cases}$ ,  $LSP_k$  is the number of paths originated from Ingress node  $k$ , and  $(\lambda_{In})_k$  is the arrival traffic of LSP  $n$  of Ingress node  $k$ .

If during  $\Delta t$ ,  $\Delta(\gamma_{ij})_k = \sum_{k=1}^{N_I} \sum_{n=1}^{LSP_k} a_{kn} \Delta(\lambda_{In})_k$  or  $\Delta\lambda_{Ik}$  has a non-zero value, the ingress node  $k$  will execute the Traffic Assignment Algorithm. There is no need for synchronization between different nodes. A small change in network state might be able to move the network to a unstable point. We show here that the algorithm moves the system to a stable point. That means that a change in the network after time  $\Delta t$  is reflected to the same node by always a smaller value.

We consider the case where the traffic belongs to only the best effort traffic . In LSP  $i$  of Ingress node  $k$ , the packet loss rate is more sensitive to the variation of the LSP traffic  $(\Delta\lambda_{Ik})_k$  than the load traffic of a link of this node, since the LSP traffic passes through all the nodes of an LSP. We have

$$S_{\lambda_i}(t) = \frac{\partial L_i(\lambda_i(t), M_i)}{\partial \lambda_i(t)} > S_{\Gamma_{ij}}(t) = \frac{\partial L_i(\lambda_i(t), M_i)}{\partial \gamma_{ij}(t)} \quad (18)$$

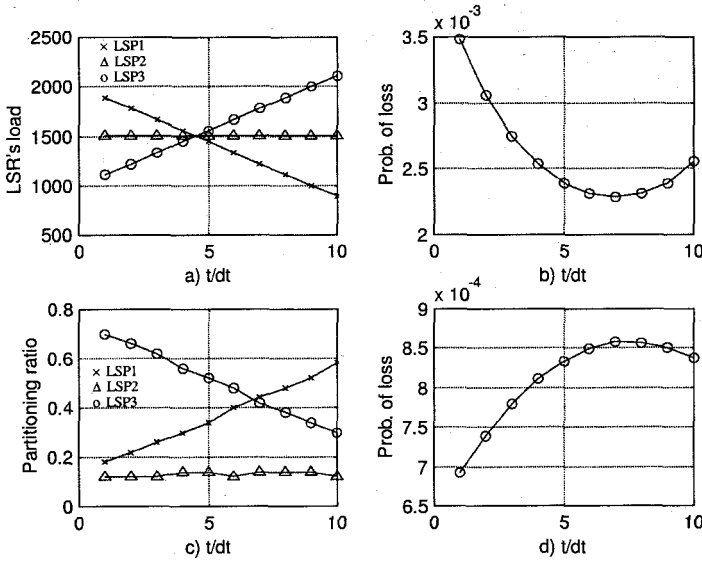
or  $\frac{S_{\Gamma_{ij}}(t)}{S_{\lambda_i}(t)} < 1$  and for the assigned traffic

$$|\Delta\lambda_i| = \left| -\frac{\sum_{j=1}^{R_i} S_{\Gamma_{ij}}(t) \Delta\gamma_{ij}(t)}{S_{\lambda_i}(t)} \right| = \left| -\sum_{j=1}^{R_i} \frac{S_{\Gamma_{ij}}(t)}{S_{\lambda_i}(t)} \Delta\lambda_i(t) \right| < \left| \sum_{j=1}^{R_i} \Delta\gamma_{ij}(t) \right|$$

This shows that the changes in the input traffic of an LSP is less than the sum of all changes in the load traffic. Changes in load traffic represents the changes in traffic partitioning of other Ingress nodes in the network. We mention here that this traffic is directed toward a link with the lowest sensitivity. That also helps to reduce the amount of the traffic variation, and to provide system stability.

## 5 Numerical Analysis and Discussion

In the following numerical analysis, we restrict attention to three parallel LSPs for the system. Each LSP is modeled by one queue with the service rate of 4 K packets/s. The queue sizes are considered to be equal to 15, 10 and 7, and the arrival rate to the ingress node is 2 K packets/s. We consider a hashing function

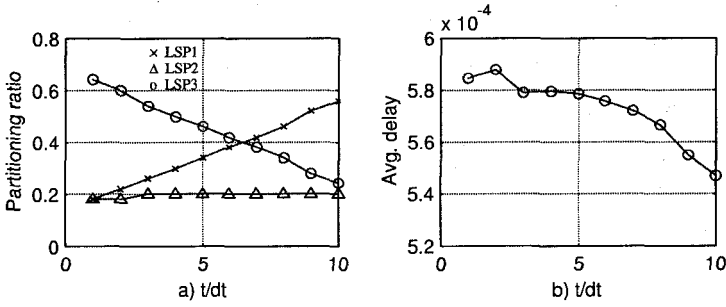


**Fig. 3.** Traffic partitioning vector for the best effort traffic, a) LSR's load,  $\lambda_{li}$ , b)  $P_b$  with equal partitioning, c) optimal partitioning vector, d)  $P_b$  with optimal partitioning.

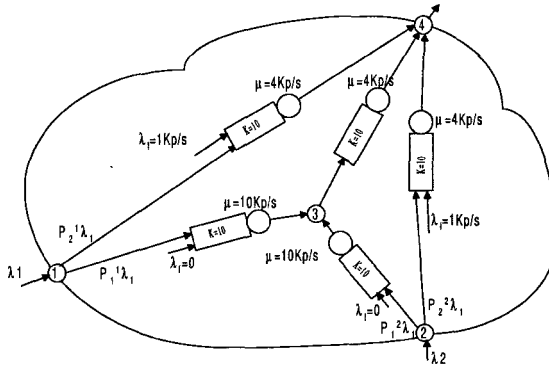
that divides the input traffic into 100 bins. Figure 3.a shows LSR's load ( $\gamma_i$ ) and their variations in time. The available capacity of LSP<sub>1</sub> increases with time, LSP<sub>2</sub> has a constant load, and LSP<sub>3</sub> is congested with time.

Figure 3.b shows the system performance measured by average  $P_b$  in the system when the input arrival is divided into three equal parts and the partitioning vector is fixed during the time. Figure 3.c shows the optimal partitioning vector in time obtained by the presented algorithm in order to minimize  $P_b$ , and Figure 3.d plots the average  $P_b$  when the optimal partitioning algorithm has been applied in the system. Plots in Figure 3 show that the partitioning vector not only depends on the system load but also depends on the queue sizes. In a general term, when the other path arrival rate of a node decreases, it can accept more traffic. Also a queue with larger size can support more traffic with the same probability of blocking. For example, consider the case when  $\gamma_1 = \gamma_2 = \gamma_3$  in figure 3.a, the vector  $P$  at this time is equal to  $P = [0.32 \ 0.14 \ 0.54]$ . It can be seen that optimal partitioning improves the system performance by more than 3 times. Figure 4 shows the same parameters when traffic is considered to be transported by TCP. The optimum partitioning vector is different from the one obtained in Figure 3. This shows that traffic assignment parameters depend not only on the network state but also on the QoS requirements. To illustrate the transient behavior of the network, an example is considered in Figure 5. For each link, as we described before, an Ingress node considers the traffic of its own established path as the *assigned traffic*, and the traffic directed by other Ingress





**Fig. 4.** Traffic partitioning vector for TCP traffic (with the arrival rates as shown in Figure 4.a), a) optimal partitioning vector, b) average delay with optimal partitioning.



**Fig. 5.** An example in order to study stability and the transient behavior of an MPLS network.

nodes to this link as the current *load traffic*. Two LSPs, whose traffic is controlled by different Ingress nodes, share the link between nodes 3 and 4. Main system parameters as service rates and arrival rates are shown in the figure. System starts with  $\lambda_1 = 3 \text{ K packets/s}$  and  $\lambda_2 = 0$  and after  $\Delta t$ , input traffic of node 2 is increased to  $\lambda_2 = 1 \text{ K packets/s}$ . The simulation results are given in Figure 6, for the Best Effort Traffic. When the arrival traffic of node 2 increases, it directs some part of the traffic to node 3. As a result, Ingress node 1 sees a change in network state and reduces the traffic directed toward node 3, which can be interpreted as a network state change for node 2. This loop continues until the system reaches to a stable point.

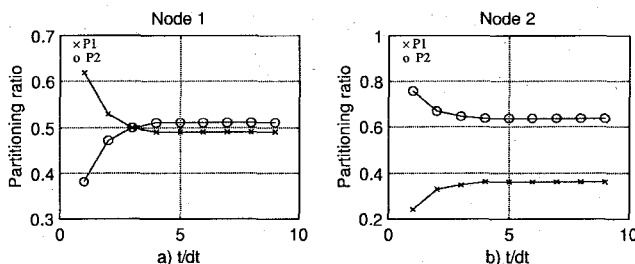


Fig. 6. Traffic partitioning parameters a) Node 1, b) Node 2.

## 6 Conclusion

This paper described an approach to network performance optimization concerning traffic engineering over MPLS. We developed an analytic model to investigate the optimal partitioning of MPLS ingress traffic into parallel label switched paths, based on the current state of the network. Each LSP was modeled by a series of queues. A stochastic framework and a partitioning algorithm were presented, which take into account the dynamics of the network state. An Algorithm was exhibited for the input traffic assignment. The system performance improvement was illustrated by numerical results. The results suggest that the optimal partitioning of input traffic increases the system performance. The efficacy of this approach depends on QoS requirements.

## References

1. E. C. Rosen, A. Viswanathan, and R. Callon, "Multiprotocol Label Switching Architecture," work in progress, Internet Draft <draft-ietf-mpls-arch-06.txt>, August 1999.
2. D. Awduche, J. Malcolm, J. Agogbua, M. O'Dell, and J. McManus, "Requirements for Traffic Engineering Over MPLS," Internet RFC 2702, Sep. 1999.
3. D. Awduche, L. Berger, D. Gan, T. Li, G. Swallow, and V. Srinivasan, "Extensions to RSVP for LSP Tunnels," work in progress, Internet Draft <draft-ietf-mpls-rsvp-lsp-tunnel-02.txt>, March 1999.
4. V. Jacobson, K. Nichols, K. Poduri, "An Expedited Forwarding PHB," RFC 2598, June 1999.
5. C. Villamizar, "OSPF Optimized Multipath (OSPF-OMP)," work in progress, Internet Draft <draft-ietf-ospf-omp-02.txt>, February 1999.
6. S. Keshav, "A Control-Theoretic Approach to Flow Control," *Proc. SIGCOM'91*, pp. 3-15, Sep. 1991.
7. W. L. Winston, *Operation Research, Application and Algorithms*, Duxbury Press, 1994.
8. I. Widjaja and A. Elwalid, "MATE: Adaptive Traffic Engineering," work in progress, Internet Draft <draft-widjaja-mpls-mate-00.txt>, August 1998.
9. T. Li, G. Swallow, and D. Awduche, "IGP Requirements for Traffic Engineering with MPLS," Internet Draft <draft-li-mpls-igp-te-01.txt>, February, 1999.

# Impact of the Ethernet Capture Effect on Bandwidth Measurements <sup>\*</sup>

Mats Björkman and Bob Melander

Uppsala University, Information Technology, Dept. of Computer Systems,  
Box 325, SE-751 05 Uppsala, Sweden.

`{Mats.Bjorkman,Bob.Melander}@docs.uu.se`

**Abstract.** In this paper we present impacts of the Ethernet capture effect on bandwidth measurements. We show that the unfairness caused by the Ethernet capture effect can have a severe impact on the ability to estimate available bandwidth. We define two metrics, surplus bandwidth and fair share bandwidth, that are protocol and application independent available bandwidth metrics. We propose a measurement method, the TOPP method, that measures link load with low risk of triggering an Ethernet capture effect. From TOPP measurements, we can estimate available bandwidth.

## 1 Introduction

Knowledge about the network bandwidth along a path from a source to a destination can be valuable for many applications that are able to adapt to changing network conditions. The bandwidth available from a source to a destination in a best effort network depends on the characteristics of the network path that the traffic travels. Not only static properties, such as the link speeds of the links forming the path, but also dynamic properties, such as competition with other traffic and changes in the routing topology are important. The dynamic properties can make the available bandwidth change dramatically over short periods of time.

In a network, we are likely to find shared-medium links such as Ethernet/802.3 links close to or local to the sender and receiver respectively. However, some of these links (notably the Ethernet/802.3), have an access method that is not fair in contention situations. This unfairness has been called the *Ethernet Capture Effect* [11]. The Ethernet capture effect makes estimation of available bandwidth hard.

In this paper we present results from experiments with an extension to the packet pair probing technique [6]. We call our probing method the *TOPP* method. The aim is to measure available bandwidth also on links with the Ethernet capture effect.

The contributions made are the following: We describe why measurement of available bandwidth is hard in Ethernet capture effect situations. We show how

---

<sup>\*</sup> This work is supported in part by the Ericsson MARCH project.

the presence of the Ethernet capture effect can be detected in measurements. We propose a measurement method, TOPP, and two metrics that can be useful when estimating available bandwidth in Ethernet capture effect situations.

The paper is organized as follows. Related work is presented in Section 2 and section 3 presents the bandwidth definitions used in the paper. The Ethernet capture effect is then described in section 4. Section 5 describes our experimental setup. Section 6 presents how the link bandwidth can be found, and section 7 shows how the Ethernet capture effect can be detected. Section 8 presents the TOPP measurement method and how we analyze TOPP measurements to estimate available bandwidth. Finally, in section 9, conclusions are drawn and future work presented.

## 2 Related Work

Paxson [10] discusses methods for measuring bottleneck bandwidth. He makes a distinction between *bottleneck* [link] *bandwidth*, which is the data rate of the slowest forwarding element of a path, and *available bandwidth*, defined as the data rate at which a connection should transmit to preserve network stability.

The packet-pair method for estimating the bottleneck bandwidth has been studied by a number of researchers [2,6]. Carter and Crovella [3] propose two packet-pair based techniques, B-probe and C-probe, to measure bottleneck link bandwidth and available bandwidth, respectively. Both techniques rely on sending trains of ICMP echo packets and applying filtering methods to the measured inter-arrival times of the reply packets.

Weaknesses of the packet pair method, such as the problem with multi-channel links, limitations due to clock resolution and out of order packet delivery are discussed by Paxson [10]. To deal with these short-comings, he introduces an extension to the packet pair technique called the PBM probing technique. In this, estimates for a range of probe train lengths (or bunch sizes) are formed and multiple bottleneck values are allowed. In [1], Allman and Paxson investigate methods for estimating the bandwidth that will be available to a new TCP connection, so that it immediately can begin sending data at that rate.

Lai and Baker [7] also suggest refinements to the packet-pair method. They propose to use a receiver-only method based on existing traffic together with a ‘potential bandwidth filtering’ scheme. In order to quickly adapt to bandwidth changes they introduce a packet window. When estimating the bandwidth, only packets that arrive within the time interval given by the window will be used.

Treno [8] is a tool that tries to measure the achievable TCP throughput between a pair of hosts. To achieve this, it simulates the full TCP algorithm including slow start. UDP packets with a suitably low time-to-live (TTL) value are sent to the destination, which will reply by sending back ICMP TTL exceeded packets. These are then interpreted as the equivalent of TCP’s acknowledgements. Since Treno simulates the TCP algorithm, it normally needs 10 seconds or more to accurately estimate the available bandwidth.

The ‘pathchar’ [5] tool estimates the latency and link bandwidth of each hop along a network path by measuring the round trip times of packets sent by a single source. Pathchar uses a TTL-based technique, similar to that of Treno, to determine how many links the probe packets should traverse. A disadvantage of pathchar is that it consumes considerable amounts of bandwidth and that it is quite slow. Downey [4] suggests techniques to improve the accuracy of pathchar’s estimates and to reduce the time required to generate them.

The Ethernet capture effect has been described in several papers [9,12,13]. Ramakrishnan and Yang [11] discuss its impact on UDP and TCP traffic and show that the capture effect degrades the performance of TCP. They propose a modified backoff algorithm, Capture Avoidance Binary Exponential Backoff (CABEB), that will overcome the capture effect.

### 3 Definitions

By *bottleneck link bandwidth* we mean the maximum transmission rate that can be achieved on the slowest link along a network path from sender to receiver. This is a well-established definition in the literature. *Available bandwidth*, on the other hand, is a less well-defined property. In this paper we have identified three metrics that can be used when characterizing the available bandwidth.

Since we do not know the details of the traffic along a network route, for instance whether traffic sources will adapt to accommodate new traffic sources (e.g. TCP), or not (e.g. UDP), we can only estimate upper and lower bounds on the bandwidth available to us. A lower bound (assuming adaptation from our side but not from the other traffic) would be the currently unused portion of the bottleneck link bandwidth. This is what we define as the *surplus available bandwidth* (or surplus bandwidth for short).

A usable upper bound on the available bandwidth would be our fair share of the bottleneck link bandwidth relative to the total traffic load offered. We define this bandwidth as the *fair share available bandwidth* (or for short fair share bandwidth). Naturally, the theoretical upper bound would be the link bandwidth. This bound could, for example, be approached by an aggressive UDP source or by a TCP source not adapting properly to congestion, when the remaining traffic consists of well-behaved TCP sources that all back off.

The actual bandwidth as perceived by a specific protocol and application should lie somewhere between the surplus bandwidth and the fair share bandwidth and this we define as the *protocol dependent available bandwidth*. In the rest of this paper we will refer to this simply as the available bandwidth.

### 4 The Ethernet Capture Effect

The Ethernet capture effect [12], is a term used to describe an undesirable side effect of the Ethernet CSMA/CD backoff algorithm. It is most apparent in a high load/few stations scenario and it results in transient or short-term unfairness. What this means is essentially that under high load, one station on a LAN can

hold on to the channel and transmit several consecutive packets even though other station(s) are contending for access.

#### 4.1 The Ethernet (802.3) Backoff Algorithm

The aim of the Ethernet medium access method is to give all stations fair access to the channel, i.e. there are no prioritized stations or classes of traffic.

Whenever a station has an Ethernet frame to send it checks if the channel is idle and in that case it attempts to transmit the frame. If other stations are trying to transmit at the same time then all stations will detect a collision. To arbitrate between the contending stations the Ethernet CSMA/CD protocol uses a backoff algorithm in which the currently offered load to the channel plays a central role. When the offered load is high, the stations should back off for longer times compared to when the load is low.

To estimate the instantaneous offered load, the stations associate with each frame a collision counter,  $n$ . It is initially zero and is then increased by one each time the frame experiences a collision. When a station has detected a collision it uses the collision counter to decide how many slot times,  $n_s$ , to wait before attempting to transmit again. This delay,  $n_s$ , is chosen as a uniformly chosen random integer in the range  $0 \leq n_s \leq 2^k - 1$  where  $k = \min(n, 10)$ . If the frame experiences 16 consecutive collisions, the retransmission procedure is aborted and the frame is discarded.

The problem with this backoff scheme is that the station which is successful in sending its frame after a collision (i.e., the station that chooses the earliest slot no other station chooses) will start with a new frame having the collision counter set to 0. The other stations involved in the collision, on the other hand, will keep their collision counter values. As a result, if the successful station is involved in a new collision with stations involved in the previous collision, the successful station will choose its random wait time in a more narrow range than the other stations. This will increase its probability of being successful again in the new collision, and that increase of probability leads to the unfairness that is called the capture effect. For every consecutive collision that is won by the same station, the probability for that station to win again will increase and quickly tend towards 1 [11].

As the number of stations that share the channel increases, the capture effect will decrease. This is because the set of stations that are involved in consecutive collisions will vary more. Since every new station in the set will have a collision counter equal to 0 (i.e., the same value as the winning station has), they too will randomize in a narrow range after an initial collision. This will then effectively decrease the probability of the same station winning many consecutive collisions.

#### 4.2 Impact on Active Probing Techniques

Most of the proposed techniques for actively measuring bandwidth are variants of the packet pair technique. There, the principle of the bottleneck spacing effect is used to estimate the bandwidth. That is, when two packets are transmitted

back-to-back from sender to receiver they will be spread out in time as they pass the bottleneck. When they exit the bottleneck this spreading will result in a time separation of the packets. That spacing between the packets is due to the transmission delay of the packets and possibly due to packets from competing traffic on the channel.

In the packet pair technique, one or several pairs of probe packets are sent from the source to the destination to measure the bottleneck or available bandwidth along that path. The bandwidth estimate,  $\tilde{b}$ , is then calculated as

$$\tilde{b} = \frac{s}{t_2 - t_1} = \frac{s}{\Delta t} \quad (1)$$

where  $s$  is the size of the probe packets and  $t_1$  and  $t_2$  are the arrival times of the two packets. Since  $s$  and the resolution by which  $\Delta t$  can be measured set a limit on the largest bandwidth that can be measured, it is desirable that  $s$  is as large as possible. Furthermore, if the interval at which the probe packets are sent,  $\Delta t_s$ , is larger than the interval corresponding to the bottleneck bandwidth, then the bottleneck spacing effect will not be noticed. For that reason, the probe packets should be sent as close to back-to-back as possible.

When the goal is to measure bottleneck *link* bandwidth, packets from competing traffic can be regarded as noise as these may compress or extend the time separation between the probe packets. To get a good estimate of the link bandwidth, that noise must be filtered out from the time measurements. If instead the goal is to measure the *available* bandwidth, the packets from the competing traffic are not noise since it is the competing traffic that makes the bottleneck available bandwidth deviate from the bottleneck link bandwidth. In order for the packet pair estimate  $\tilde{b}$  to accurately estimate the available bandwidth, it is important that packets from the competing traffic interleave with the probe packets in proportion to the channel load. However, this requires that the medium access method is fair.

Since the Ethernet capture effect alters the fairness of access to the channel it also affects the packet pair probing techniques. When measuring bottleneck *link* bandwidth it will work to our advantage since the probe packets are likely to block out packets from the competing traffic. However, when measuring *available* bandwidth the capture effect will affect the packet pair techniques negatively. This is because the packets from the competing traffic will not interleave with the probe packets in proportion to the load on the channel, due to the blocking nature of the capture effect.

**C-probe** Being a variant of the packet pair probing technique, C-probe [3] will be affected by the capture effect for the same reasons as explained above. In C-probe, the goal is to measure available bandwidth and this is done by sending trains of ICMP echo packets to the destination host. The probe packets (i.e., the ICMP packets) in the train are of equal size and are sent as close to back-to-back as possible.

The available bandwidth calculation is based on equation (1), when  $t_1$  and  $t_2$  are the arrival times of the *first* and the *last* packet in the train. As the probe

packets are sent as close to back-to-back as possible, a queue will quickly build up in a contention situation. Now, because of the capture effect there is a high probability that several packets in that queue will traverse the channel without being interleaved with packets from competing traffic. As a consequence, the bandwidth estimate given by equation (1) will be too high.

## 5 Experimental Setup

All the results we present in this paper are based on experiments done on a 10 Mbps Ethernet LAN, using a number of Sun Sparcstations 4 running Solaris 2.6 and a number of Intel-based PC's running Linux (kernel versions 2.0.35 and 2.2.3). The Sparcstations are all equipped with an Am7990 (Lance) based Ethernet interface whereas the different Linux hosts use a variety of network interfaces (3Com Etherlink III, D-Link DFE-530TX and SMC EtherEZ).

## 6 Finding the Link Bandwidth

We want to estimate the link bandwidth of the bottleneck link. Packet pair probing is a technique that can be used for this. Instead of using a number of pairs, we extend the ideas of B-probe [3] and send a long train of back-to-back packets through the network. Here, the Ethernet capture effect can actually help us when trying to establish the link bandwidth of the bottleneck link. Should the bottleneck link be an Ethernet, the Ethernet capture effect will affect our bandwidth measurements as has been explained earlier. Using a long train of packets will increase the possibility that we will see a capture effect if there is one. When analyzing packet pair inter-arrival times, times reflecting the link bandwidth will (as explained earlier) be over-represented as compared to a link without any capture effect.

As can be seen in figure 1, packets from a train moving through a saturated Ethernet will exhibit an extremely bursty behavior, with several long bursts at, or near in the case of contention slots passing, link bandwidth speed. Should this Ethernet be the bottleneck link, due to the capture effect the link bandwidth can many times be estimated without need for advanced filtering (in the example in figure 1, a median value would be sufficient).

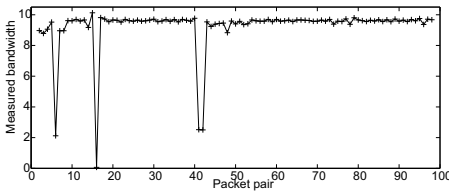
## 7 Detecting the Ethernet Capture Effect

On a shared-medium link with (short-term) fair resolution of contention, the probability of bursts of frames at link speed will rapidly decrease with the burst length. Consider two stations competing for the medium. With a fair contention resolution, the probability of a certain station winning one collision would be 0.5, the same station winning two successive collisions would be 0.25, three successive collisions 0.125 etc., yielding a mean expected burst length of 2.

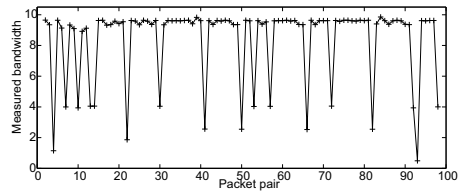


On an Ethernet, however, the capture effect will favor the station already sending when this station has a queue of frames to send. Once a competing station has started losing contention resolutions against a station with a queue of frames, chances are (as explained earlier) that it will continue to lose as long as the winning station has frames queued up for sending.

In fact, on a saturated Ethernet with only two senders, once a station has started losing, the sum of probabilities that it will get to send the frame at all before giving up (after 16 successive collisions) can be calculated to less than 50%. (The sum of probabilities that the loser of collision 1 will win any of the collisions 2, 3, ... up to 16 amounts to between 40% and 50%, dependent on the number of ties at contention resolutions). Thus, on a saturated Ethernet with few sending stations, the probability of seeing long bursts will be enormously increased as compared to a fair medium (above). In the saturated two-station example, once queues have built up, the median burst length (counting bursts of size 1 and longer) is 16, since the probability is less than 50% that the loser will be able to break in during exponential backoff, but once the losing station discards the losing frame, its next frame will have a fresh (zero) collision counter, and the losing station stands a 50% chance to win the 17th collision and thus end the burst from the other station. See also the sample train in figure 1.



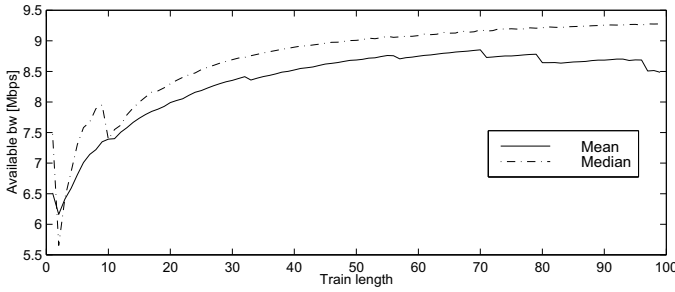
**Fig. 1.** Packet bursts for packets from the probing station. 10 Mbps link, one station is generating 7 Mbps of CBR cross traffic.



**Fig. 2.** Packet bursts for packets from the probing station. 10 Mbps link, six stations are together generating 7 Mbps of CBR cross traffic.

Hence, we can detect the Ethernet capture effect by examining the distributions of burst lengths. In figure 1, we saw a typical example of burst lengths when we have two competing stations. As the number of competing stations increase, the capture effect (in the sense that one and the same sender gets to send again and again) will decrease in significance [1]. However, still at seven competing stations (figure 2), we see this over-representation of burst lengths, albeit with shorter burst lengths than in the two-station case shown in figure 1.

The conclusion from this section is the following: as long as we have an Ethernet capture effect significant enough to cause an over-representation of long link bandwidth bursts (effectively destroying our possibility to use packet trains to measure available bandwidth), this over-representation can be detected by examining the burst length distribution. Once we have detected that we indeed



**Fig. 3.** Measured available bandwidth using packet trains. One station is generating 7 Mbps of CBR cross traffic.

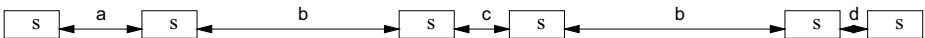
have a capture effect situation, we cannot use a train method, but must use other methods to estimate available bandwidth, see below.

## 8 Measurements in an Ethernet Capture Effect Situation

As explained above, packet trains will over-estimate available bandwidth in a capture effect situation. Figure 3 shows the available bandwidth as estimated by packet trains on a 10 Mbps Ethernet. The single cross-traffic source offers 7 Mbps, so surplus bandwidth is 3 Mbps. The fair share bandwidth at the 10 Mbps offered by the probe source would be  $10/(7+10) \times 10 = 5.9$  Mbps. As can be seen, the packet train method over-estimates available bandwidth far beyond the fair share bandwidth. Hence, we must instead try to estimate traffic characteristics without triggering any capture effect.

### 8.1 Measuring Without Triggering the Capture Effect

The long bursts typical for the Ethernet capture effect occur when the sender has a queue of frames to transmit. Hence, we should measure bandwidth without causing queues to build up. One way to avoid this is to measure bandwidth by sending separated packet pairs, where the pairs are so well separated that any disturbances to the network caused by one pair (e.g. queues caused by temporary overloading of the link) have been normalized before the next pair is sent.



**Fig. 4.** Parts of a TOPP train. The equally sized probe packets  $s$  are sent as pairs with decreasing intra-pair spacing, i.e,  $d < c < a$ . The spacing between the pairs are chosen randomly in the range  $[b - \epsilon, b + \epsilon]$  where  $\epsilon$  is a small number. For each intra-pair spacing,  $a$ ,  $c$ , etc,  $n$  pairs are sent.

Instead of directly trying to measure available bandwidth, we try to find the surplus bandwidth of the bottleneck link. From that and from knowledge of the link bandwidth (measured e.g. using methods described earlier), we can estimate lower and upper bounds of available bandwidth using the methods explained above.

## 8.2 The TOPP Measurement Method

When trying to estimate the surplus bandwidth, we send a sequence of packet pairs, where the pairs are well separated in order to minimize the Ethernet capture effect. The intra-pair spacing of the packet pairs is decreased in known steps in order to momentarily offer different loads. In the presented measurements we have used steps of 1 Mbps, i.e. the first run of pairs have an intra-pair spacing corresponding to a momentary load of 1 Mbps, the next to 2 Mbps etc. We call this method the *TOPP* method (short for Trains Of Packet Pairs). Figure 4 illustrates this.

Using the TOPP method, we can analyze the effects of a particular offered load without causing long-term disturbances to the network (e.g. without triggering any long-term Ethernet capture effects), disturbances that had been caused had we actually offered that same load for any longer period.

Since the TOPP method is independent of the inter-pair spacing, this spacing can be large for links loaded close to saturation in order to minimize the impact of probe traffic. As an example, assume an inter-pair spacing of 100 ms. For a momentary load of 5 Mbps, we send two frames (each takes 1.1 ms to send) with a 1.1 ms intra-packet space and then wait 100 ms before the next pair. The real load on the link would then be  $(1.1 + 1.1)/(1.1 + 1.1 + 1.1 + 100) = 0.022 = 2.2\%$  of 10 Mbps, i.e. 210 kbps, while we can estimate the effects of a 5 Mbps load.

For the TOPP measurements presented below, the inter-pair spacing has been 20 ms, giving real loads of 670 to 970 kbps while offering momentary loads of 1 to 10 Mbps. Since our experiments have controlled cross-traffic, we know that this spacing will not cause link saturation. For measurements on links with unknown traffic, we recommend longer inter-pair spacing.

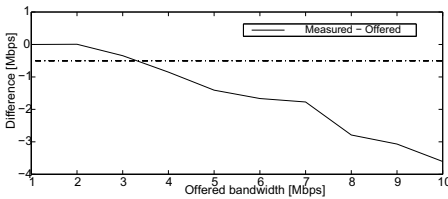
By using the TOPP method of “measuring load without loading”, we can find the surplus bandwidth of an Ethernet by analysis of TOPP reception times as the next section describes.

## 8.3 Analysis

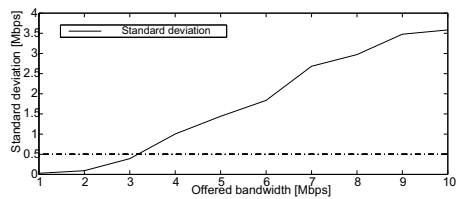
When analyzing TOPP reception times, we have used two interesting metrics.

First, as long as our offered load does not saturate the link, we will see a measured bandwidth that is close to the offered. As the link saturates, i.e. our offered bandwidth has reached the surplus bandwidth, the measured bandwidth will start deviating from the offered bandwidth. This point can, in most measurements we have performed, easily be detected. An example is seen in figure 5.

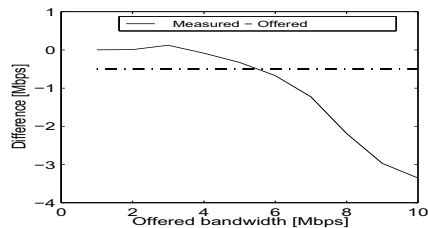
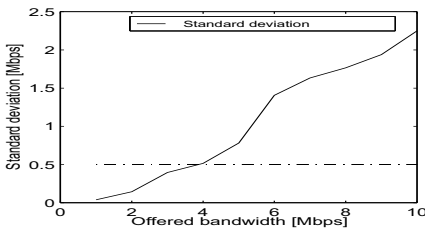
Second, as we cross the link saturation point, the standard deviation in our measurements will increase rapidly, since on a saturated link we will see com-



**Fig. 5.** TOPP measured surplus bandwidth using the *measured - offered* metric. 10 Mbps link, 7 Mbps of CBR cross traffic.



**Fig. 6.** TOPP measured surplus bandwidth using the *standard deviation* metric. 10 Mbps link, 7 Mbps of CBR cross traffic.



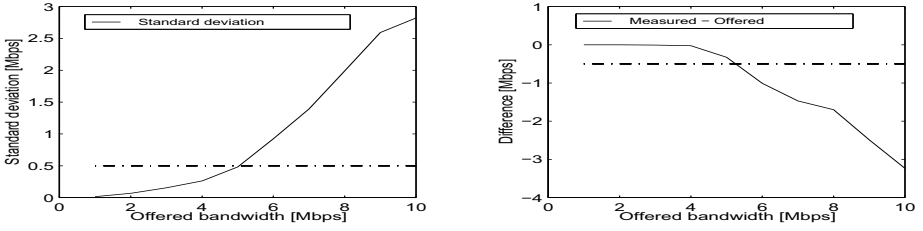
**Fig. 7.** TOPP measured surplus bandwidth using both metrics. 5 Mbps of VBR cross traffic.

peting traffic at virtually every slot, causing our measured bandwidth to either jump up (should our first frame be delayed and both probe frames then be sent more closely spaced than offered), or jump down (should one or more competing frames be sent in between our probe frames), but seldom would both probe frames cross the link undisturbed and the measured bandwidth be the offered. As seen in the example in figure 6, there is a sharp increase in standard deviation as we reach link saturation.

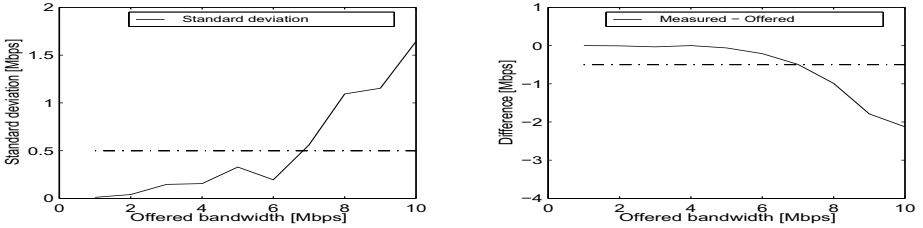
Below we present figures of the measured and offered bandwidth difference and the bandwidth standard deviation for measurements with varying cross traffic bandwidths and cross packet sizes. We have used a 0.5 Mbps difference as the limit where we consider saturation to have occurred, and a 0.5 Mbps standard deviation as the standard deviation limit. As shown below, the surplus bandwidth can in most of our measurements be estimated quite accurately.

Figure 7 shows results from a measurement with varying cross-traffic frame sizes (as opposed to the fixed size CBR cross-traffic from figures 5 and 6). In this measurement, the offered cross traffic is 5 Mbps. As can be seen, both measures seem to give good estimations of the surplus bandwidth.

Figure 8 shows estimations when having 5 Mbps CBR cross traffic, and figure 9 estimations for 3 Mbps CBR cross traffic. For both these cross-traffic bandwidths, our measures give good estimations of surplus bandwidth.



**Fig. 8.** TOPP measured surplus bandwidth using both metrics. 5 Mbps of CBR cross traffic.



**Fig. 9.** TOPP measured surplus bandwidth using both metrics. 3 Mbps of CBR cross traffic.

With knowledge of the surplus bandwidth and link bandwidth, we can calculate the cross-traffic bandwidth. Knowing this, we can for a given offered load calculate what our fair share of the link bandwidth would be.

For example, if the surplus bandwidth is 3 Mbps and the link bandwidth is 10 Mbps, we know that cross traffic loads the link with 7 Mbps. Should we offer 7 Mbps ourselves, our fair share of the 10 Mbps link would be  $(7/(7+7)) * 10 = 5$  Mbps. Hence, should our application initially offer 7 Mbps to this link, our perceived available bandwidth would likely be not less than the surplus bandwidth of 3 Mbps and not more than the fair share bandwidth of 5 Mbps.

## 9 Conclusions and Future Work

From the work presented in this paper, we draw several conclusions:

When estimating available bandwidth, two reasonable protocol-independent lower and upper bounds are surplus and fair share bandwidth, respectively.

Trains of back-to-back packets cannot be used to accurately measure available bandwidth in Ethernet capture effect situations. This means that methods like C-probe do not give accurate estimations in these situations, rather will they over-estimate the available bandwidth.

Burst length analysis can detect an Ethernet capture effect situation.

The presented TOPP method of injecting well-separated packet pairs with decreasing intra-pair spacing can be used to measure bandwidth without causing long-term disturbances to the link.

When estimating surplus bandwidth, the difference between offered and measured bandwidth, and the standard deviation of the measured bandwidth, are two metrics that are likely to be useful in the analysis.

Future work include multi-hop measurements and the adaptation of our methods to other shared-media links such as wireless links.

## References

1. Mark Allman and Vern Paxson. On Estimating End-to-End Network Path Properties. In *SIGCOMM '99 Conference Proceedings*, pages 263–273, Cambridge, MA, USA, August 31–September 3, 1999. ACM SIGCOMM Computer Communication Review, 29(4).
2. Jean-Chrysostome Bolot. End-to-end packet delay and loss behavior in the internet. In *SIGCOMM '93 Conference Proceedings*, pages 289–298, San Francisco, CA, USA, September 13–17, 1993. ACM SIGCOMM Computer Communication Review, 23(4).
3. Robert L. Carter and Mark E. Crovella. Measuring bottleneck link speed in packet-switched networks. Technical Report TR-96-006, Boston University Computer Science Department, Boston, MA, USA, March 1996.
4. Allen B. Downey. Using pathchar to estimate Internet link characteristics. In *SIGCOMM '99 Conference Proceedings*, pages 241–250, Cambridge, MA, USA, August 31–September 3, 1999. ACM SIGCOMM Computer Communication Review, 29(4).
5. Van Jacobson. Pathchar - a tool to infer characteristics of Internet paths. Presented at the Mathematical Sciences Research Institute (MSRI); Slides available from <ftp://ftp.ee.lbl.gov/pathchar/>, April 1997.
6. Srinivasan Keshav. A control-theoretic approach to flow control. In *SIGCOMM '91 Conference Proceedings*, pages 3–15, Zürich, Switzerland, September 3–6, 1991. ACM SIGCOMM Computer Communication Review, 21(4).
7. Kevin Lai and Mary Baker. Measuring bandwidth. In *Proceedings of IEEE INFOCOM '99*, New York, USA, March 21–25, 1999.
8. Matt Mathis and Jamshid Mahdavi. Diagnosing internet congestion with a transport layer performance tool. In *INET '96*, Montreal, Canada, June 24–28, 1996.
9. Mart L. Molle. A New Binary Logarithmic Arbitration Method for Ethernet. Technical Report CSRI-298, Computer Systems Research Institute, University of Toronto, Toronto, Canada, April 1994.
10. Vern Paxson. End-to-end internet packet dynamics. In *SIGCOMM '97 Conference Proceedings*, pages 139–152, Cannes, France, September 14–18, 1997. ACM SIGCOMM Computer Communication Review, 27(4).
11. K. K. Ramakrishnan and Henry Yang. The Ethernet Capture Effect: Analysis and Solution. In *Proceedings of IEEE 19th Conference on Local Computer Networks*, MN, USA, October, 1994.
12. Rich Seifert. The Effect of Ethernet Behaviour on Networks using High-Performance Workstations and Servers. Technical Report, Networks and Communications Consulting, Cupertino, CA, USA, March 1995.
13. Brian Whetten, Stephen Steinberg and Domenico Ferrari. The Packet Starvation Effect in CSMA/CD LANs and a Solution. University of California at Berkeley.

# Statistical Study of the Correlation Between Topology and Wavelength Usage in Optical Networks with and without Conversion

Christian Fenger<sup>1</sup>, Emmanuel Limal<sup>2</sup>, Ulrik Gliese<sup>2</sup>, and Cathal J. Mahon<sup>1</sup>

<sup>1</sup> Tele Danmark, Telegade 2, DK-2630 Taastrup, Denmark  
{cfeng,cmah}@tdk.dk

<sup>2</sup> Research Center COM, Technical University of Denmark, DK-2800 Lyngby,  
Denmark  
{el,ug}@com.dtu.dk

**Abstract.** We present a statistical study of wavelength usage in relation to topology characteristics for optical WDM networks with static traffic requirements, where the traffic is routed both with and without wavelength conversion. We identify new general correlations between parameters describing network topologies and wavelength usage. We find that the regularity of a network and the number of spanning trees in a network are accurate and reliable measures of the routing efficiency in terms of wavelengths used to accommodate the required traffic. An empirical formula is given for the average number of wavelengths required to accommodate the traffic as a function of the number spanning trees in the network. We observe that in most cases, the wavelength usage with and without wavelength converters are identical. Consequently, the correlations between network topology and traffic assignment efficiency are true for both types of networks.

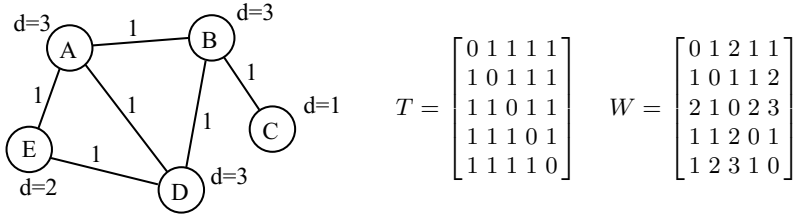
## 1 Introduction

Optical communication networking is a rapidly growing business. All over the world, many new carriers are investing to deploy completely new networks based on the newest optical fiber technologies. Due to the need for higher bandwidth, the trend moves towards all-optical networks where knowledge of wavelength routed networks becomes increasingly important. Despite this increasingly competitive situation, a general knowledge base on which to build the design and planning of optical networks is still somewhat limited. Up until now, studies of the correlation between network topologies and efficiency in accommodating a certain traffic demand have been based on specific network configurations instead of systematic surveys [1]-[4]. In this new work, we randomly generate a large number of networks of different topologies and study statistically how well the traffic is accommodated in networks with and without wavelength converters as a function of a number of well defined parameters describing the network topology. We can thereby generate a general knowledge base for network designers.

In the next section we define the problem and describe the routing and wavelength assignment algorithms used for networks with and without wavelength converters. In Section 3 we present the results for networks without wavelength converters. In Section 4 we show and discuss results for networks with wavelength converters and compare these with the results from Section 3.

## 2 Problem and Algorithm Description

The goal of this work is to obtain a general understanding of how different WDM network topologies affect the number of wavelengths required to accommodate a given traffic demand. This is achieved by randomly generating a large number of networks (several million) for given numbers of nodes and links. The wavelength usage is evaluated for each network as a function of the average and variance of the node degree and the number of spanning trees as described in detail in Section 3. The degree,  $d$ , of a node is the number of links incident on the node, as shown in Figure 1. General statistical results are then obtained by averaging over all generated networks for each of these parameters. For each topology we



**Fig. 1.** Example of a network with 5 nodes and 6 links. The weight is one between physical connections. The degree is given for each node. The average node degree is 2.4, and the variance of the node degree is 0.8. The network traffic,  $T$ , and link weight,  $W$ , matrices are also shown.

assume one fiber per link, no upper limit on the number of wavelengths, uniform traffic, uniform link weight, and bi-directional links. The element of the traffic matrix,  $T(i, j)$ , gives the traffic between node  $i$  and node  $j$  in terms of number of wavelengths while the element of the link weight matrix,  $W(i, j)$ , indicates the number of hops in going from node  $i$  to node  $j$ . As an example, consider the network in Figure 1 for which the weight matrix is determined by using the shortest path connecting each node pair.

### 2.1 The Algorithm for Networks Without Wavelength Converters

The algorithm for networks without wavelength converters contains basically three steps as described below. Due to the huge number of random networks



generated (several million), we have carefully chosen well-known and efficient algorithms of low complexity. The routing algorithm is an algorithm by Floyd [5] that calculates the shortest path of all node pairs with complexity  $\mathcal{O}(N^3)$ , where  $N$  is the number of nodes in the network. When all routes are known, the wavelength assignment problem is modified into a graph colouring problem. It is done by creating a graph for the path collisions such that each node of this graph corresponds to a path and two nodes are adjacent if their corresponding paths share at least one link in the network topology. The obtained path collision graph must then be coloured such that no two adjacent nodes have the same colour. We chose to solve this problem by using a well-known graph colouring heuristic algorithm called the *largest-first* or the *descending-order-of-degree* algorithm [6]. Since the path collision graph has  $N(N-1)/2$  nodes, where  $N$  is the number of nodes in the network, then complexity of the *largest-first* algorithm is  $\mathcal{O}(N^4)$ . This algorithm basically allocates the lowest number ed colour to the nodes of larger degree first. To ensure that the results for wavelength usage for networks without wavelength converters do not depend on the graph colouring algorithm efficiency, the *largest-first* algorithm has been compared to the more complex, slower by generally better performing, *Dsatur* graph colouring algorithm [7]. Results are discussed in section 3.

The steps in the algorithm for no wavelength conversion are:

- Step 1:* A connected network is randomly generated for a given number of nodes and links.
- Step 2:* The paths between all node pairs are determined using a shortest path algorithm [5].
- Step 3:* Wavelengths are assigned by modifying the wavelength assignment problem into the graph colouring problem under the restriction that two identical wavelengths cannot share the same link and no wavelength conversion can take place. The graph is coloured using a graph colouring algorithm.

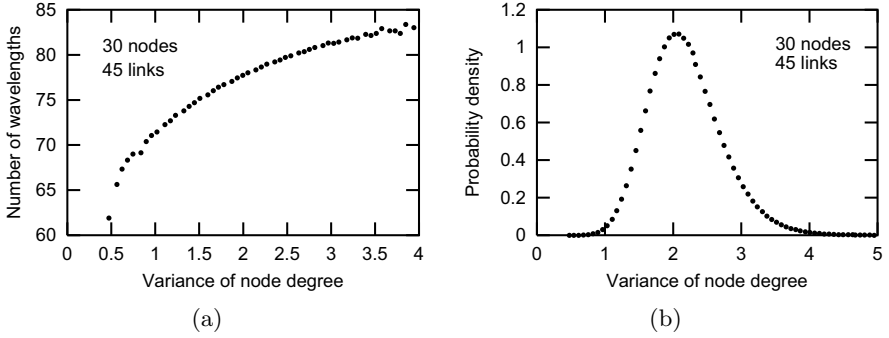
## 2.2 The Algorithm for Networks with Wavelength Converters

For networks with wavelength converters, the algorithm simply contains step 1 and 2 of the algorithm presented in the previous section as no constraints are placed on the wavelength assignment. The wavelengths are literally chosen on a link to link basis.

## 3 Results and Discussion for No Conversion

Efficient network planning is a complex challenge and knowledge of any general trends in network performance in relation to network topology is desirable as it can reduce planning complexity. Therefore, network planners need reliable parameters to characterize network topologies.

The variance of the node degree is one such parameter as it gives a simple measure of the network regularity (i.e. the variance of the node degree) and



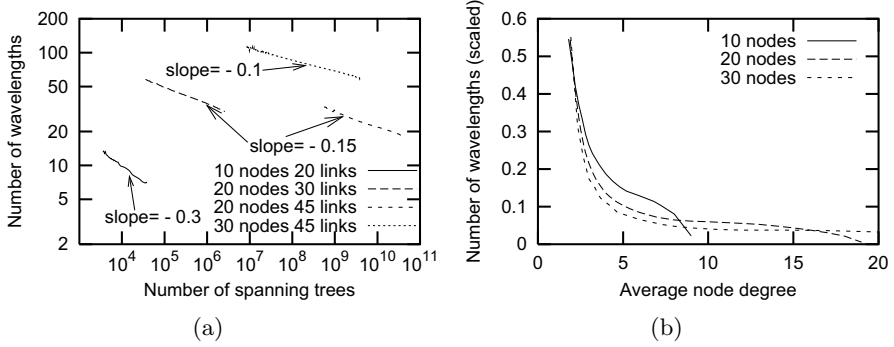
**Fig. 2.** (a) Average usage of wavelengths as a function of the variance of the node degree. (b) Probability density of the variance of the node degree. Notice that the distribution is discrete, due to the limited number of possible variances of node degree. Data from 4 million networks are used for the plots.

complements the information given by the average node degree. In Figure 2(a) we have plotted the average number of wavelengths required to accommodate the traffic demand as a function of the variance of the node degree. As seen, the number of wavelengths used grows with the variance, and the network uses a minimum of wavelengths when it is regular (all nodes are of the same degree). To see how many topologies satisfy the low variance requirement, we have also generated the probability density of the variance of the node degree, as shown in Figure 2(b). As seen, very few topologies have very low variance of node degree. This suggests that the number of calculations performed by network optimization algorithms can drastically be reduced.

Another important and more informative parameter for characterizing the topology of an optical network is the number of spanning trees in the network. A tree is a subnetwork of a network, which does not contain any loops, and a spanning tree of a network is a tree containing all nodes of the network. The network shown in Figure 1 contains eight spanning trees. The number of spanning trees is a measure of the number of possible routes between nodes in the network. It easily identifies the restriction in the number of routes due to links that disconnect the network when removed (cut-edges). The calculation of the number of spanning trees,  $S$ , in a network is easily obtained by calculating the determinant of the product of the matrix  $A$  and its transposed matrix  $A^t$  where  $A$  is an incidence matrix of a directed graph obtained by assigning orientations to the edges of the network topology graph such that no self-loops appears with one row deleted [8]:

$$S = \det(AA^t)$$

In Figure 3(a) we show the average number of wavelengths required as a function of the number of spanning trees in the network. The simulation points form straight lines for two orders of magnitude, on a double-log scale, indicating a



**Fig. 3.** (a) Average usage of wavelengths as a function of the number of spanning trees. (b) Average usage of wavelengths, scaled with the total traffic, as a function of the average node degree.

power-law. A comparison of the curves shows that the exponent does not depend on the number of links, but only on the number of nodes, and is approximately given by  $-3/N$ .

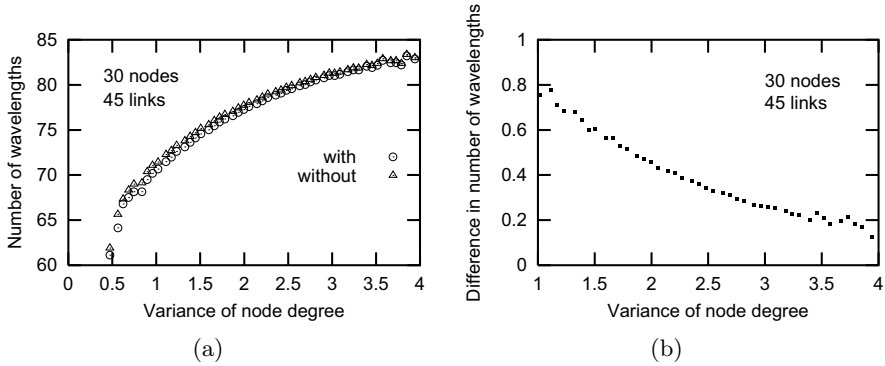
The equation below gives the dependency of the average wavelength usage,  $\langle \lambda \rangle$ , on the number of nodes,  $N$ , the number of links,  $L$ , and the number of spanning trees,  $S$ .

For large networks (more than 10 nodes) with a modest average node degree (3-6) we therefore get an empirical law for the average use of wavelengths as function of the number of spanning trees,  $S$ , when the number of nodes,  $N$ , and the number of links,  $L$ , are held constant:

$$\langle \lambda \rangle = \alpha(N, L) \cdot S^{-3/N}$$

Here  $\alpha$  is an increasing function of  $L$  and a slowly decreasing function of  $N$ . For very small networks or very strongly or sparsely connected networks the universal feature expressed by the power-law is disturbed. As the number of nodes rises (while the number of links is held constant)  $\alpha$  drops, and as the number of links rises (while the number of nodes is held constant)  $\alpha$  rises. From this general equation the network planner can get an insight into the wavelength usage of the type of topology to which a specific network belongs.

The average node degree is also a relevant parameter for the network planners. In Figure 3(b) we see the relationship between the average node degree and number of wavelengths required to satisfy the traffic demand. To easily compare the results for different network sizes, the number of wavelengths is scaled with the total traffic demand. It is seen that the gain from raising the average node degree from small values has a significant impact on the decrease in wavelength usage, whereas a raise from larger values only has a small effect. This change in gain from raising the average node degree occurs around an average node degree of 4-5 for all networks, indicating that these are near optimum values. In the



**Fig. 4.** (a) Average usage of wavelengths as a function of the variance of the node degree with wavelength converters and without wavelength converters. (b) Difference in average usage of wavelengths between traffic with and without wavelength converters as a function of the variance of the node degree.

following section, we investigate wavelength usage for networks with wavelength converters as a function of the same topology parameters, namely the variance of the node degree and the number of spanning trees. Second it is seen that the performance of the algorithm actually rises with the size of the network. Where the performance is measure as the number wavelengths used scaled with the total traffic when the average node degree is held constant. As the total traffic scales with square of the number of nodes and the number of links for a constant average degree scales with the number of nodes, then the number of wavelengths divided by the total traffic should be constant in a network, which does not scale. Here we see that the traffic is more efficiently accommodated for larger networks.

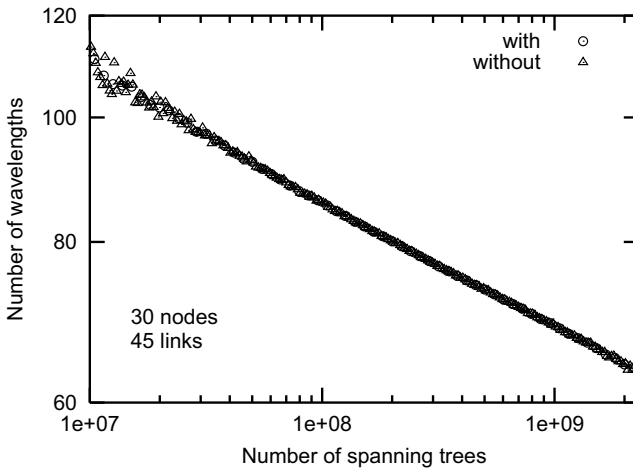
## 4 Comparison with Wavelength Conversion

The statistical results estimated over millions of networks show that the correlation between topologies and wavelength usage is exactly similar for networks with and without wavelength converters. However, small differences in wavelength usage were noticed and are discussed in this section. In Figure 4(a) the average usage of wavelength is plotted for networks with and without wavelength converters as a function of the variance of node degree for networks of 30 nodes and 45 links. Although it reduces the blocking probability and reduces management complexity for dynamic traffic assignment it has no (or only a very small) effect on static routing. The curves indicate a very small difference in the number of wavelengths required to assign the traffic. The difference, when it exists, is generally of the order of one or two wavelengths which represents, on average for all networks, an increase in wavelength usage of only 0.5%.

Contrary to what was expected, the difference in wavelength usage increases for networks with small variances (regular networks) as shown in Figure 4(b).

We believe that this is mainly due to the increase in the inefficiency of the *largest-first* graph colouring algorithm as the networks become more regular. To verify this assertion, we have in a provisional study compared the *largest-first* algorithm to the more complex graph colouring algorithm, *Dsatur* [7], in which the *Dsatur* algorithm proved better than the *largest-first* by sometimes using fewer wavelengths in the few cases where the network with wavelength converters had a lower wavelength usage. This matter is, however, still under investigation. See also [9] for a comparison between different heuristic methods of graph colouring.

In Figure 5, the wavelength usage for networks with and without wavelength converters is plotted versus the number of spanning trees of the network topologies. Results for networks without wavelength converters are shown for the *largest-first* graph colouring algorithm. It is clearly seen that the analytical formula of section 3 is still valid for networks with wavelength converters.



**Fig. 5.** Average usage of wavelengths as a function of the number of spanning trees with wavelength converters and without wavelength converters.

## 5 Conclusions

This paper contains a thorough investigation of the influence of network topology on wavelength usage. This investigation was done through an exhaustive statistical study of randomly generated topologies for networks with and without wavelength converters. We define general parameters that accurately describe network topology properties and clearly identify the correlation between the topology properties and the wavelength usage in networks. The correlations are exactly identical for networks with and without wavelength converters. It is

shown that for a given number of nodes there are only very few topologies that perform efficiently in terms of wavelength usage and we indicate the criteria that identifies them. These criteria can be used to simplify the optimization process during network planning. We have shown that, in general, networks that have an average node degree of 4-5 and a low node degree variance are among the best possible in terms of wavelength usage. Furthermore, we have clearly identified the number of spanning trees as a very accurate measure of the quality of a network in terms of traffic accommodation efficiency, and we have derived a simple equation for the average number of wavelengths required for a given topology as a function of the number of the nodes, links, and spanning trees it contains. In conclusion, we present a statistical analysis of the general network behavior in terms of wavelength usage over a very large number of network topologies and obtain general trends as well as a formula that can drastically narrow down the number of solutions that need to be considered when designing a network.

## References

1. Tan, T.K. and Pollard, J.K.: 'Determination of minimum number of wavelengths required for all-optical WDM networks using graph coloring', *Electronics Letters*, Vol.31, pp.1895-1897, October 1995.
2. Subramaniam, S., Barry, R., 'Wavelength Assignment in Fixed Routing WDM Networks', *IEEE Int. Conf. on Comm.*, Vol.1, pp.406-410, 1997.
3. Mokhtar, A., Aziöglu, M., 'Adaptive Wavelength Routing in All-Optical Networks', *IEEE/ACM Trans. Networking*, Vol.6, pp.197-206, April 1998.
4. Baroni, S., Bayvel, P., Gibbens, R.J., and Korotky, S.K., 'Analysis and Design of Resilient Multifiber Wavelength-Routed Optical Transport Networks', *J. Lightwave Technol.*, Vol.17, pp.743-758, May 1999.
5. Floyd, R.W.: 'Algorithm 97 Shortest Path', *Communications of the ACM*, Vol.5, p.345, 1962.
6. Welsh, D.J.A. and Powell, M.B., 'An upper bound for the chromatic number of a graph and its application to timetable problems', *The Computer Journal*, Vol.10, pp.85-86, 1967.
7. Brelaz, D., 'New Methods to color the vertices of a graph', *Communications of the Association for Computing Machinery*, pp.251-256, 1979.
8. Thulasiraman, K. and Swamy, M., *Graphs: Theory and Algorithms*, John Wiley & Sons, 1992.
9. Parkinson, E. and Warren, P.R., 'A comparison of graph colouring techniques', *South African Comp. J.*, Vol.14, pp.14-19, 1995.

# Computing Blocking Probabilities in Multi-class Wavelength Routing Networks \*

Sridhar Ramesh, George N. Rouskas, and Harry G. Perros

Department of Computer Science, North Carolina State University

**Abstract.** We present an approximate analytical method to compute efficiently the call blocking probabilities in wavelength routing networks with multiple classes of calls. The model is fairly general and allows each source-destination pair to serve calls of different classes, with each call occupying one wavelength per link. Our approach involves two steps. The arrival process of calls on some routes is first modified slightly to obtain an approximate multi-class network model. Next, all classes of calls on a particular route are aggregated to give an equivalent single-class model. Thus, path decomposition algorithms for single-class networks may be extended to the multi-class case. Our work is a first step towards understanding the issues arising in wavelength routing networks that serve multiple classes of customers.

## 1 Introduction

A basic property of single mode optical fiber is its enormous low-loss bandwidth of several tens of Terahertz. However, due to dispersive effects and limitations in optical device technology, single channel transmission is limited to only a small fraction of the fiber capacity. To take full advantage of the potential of fiber, the use of wavelength division multiplexing (WDM) techniques has become the option of choice, and WDM networks have been a subject of research both theoretically and experimentally [1]. Optical networks have the potential of delivering an aggregate throughput in the order of Terabits per second, and they appear as a viable approach to satisfying the ever-growing demand for more bandwidth per user on a sustained, long-term basis.

The wavelength routing mesh architecture appears promising for Wide Area Networks (WAN). The architecture consists of wavelength routers interconnected by fiber links. A wavelength router is capable of switching a light signal at a given wavelength from any input port to any output port. A router may also be capable of enforcing a shift in wavelength [2], in which case a light signal may emerge from the switch at a different wavelength than the one it arrived. By appropriately configuring the routers, all-optical paths (lightpaths) may be established in the network. Lightpaths represent direct optical connections without any intermediate electronics. Because of the long propagation delays, and the time required to configure the routers, wavelength routing WANs are expected

---

\* This work was supported by the NSF under grant ANI-9805016.

to operate in circuit-switched mode. This architecture is attractive for two reasons: the same wavelength can be used simultaneously at different parts of the network, and the signal power is channeled to the receiver and is not spread to the entire network. Hence, wavelength routing WANs can be highly scalable.

Given the installed base of fiber and the maturing of optical component technology, it appears that current network technologies are transitory, and will eventually evolve to an all-optical, largely passive infrastructure. A feasible scenario for near-term large-scale all-optical networks has emerged in [1], and it is envisioned that wavelength routing WANs will act as the backbone that provides interconnection for local area photonic sub-networks attached to them. The contribution of our work is the development of an approximate analytical framework for evaluating the performance of multi-class wavelength routing networks.

The problem of computing call blocking probabilities under static routing with random wavelength allocation and with or without converters has been studied in [3,4,5,6,7,8]. The model in [3] is based on the assumption that wavelength use on each link is characterized by a fixed probability, independently of other wavelengths and links, and thus, it does not capture the dynamic nature of traffic. In [4] it was assumed that statistics of link loads are mutually independent, an approximation that is not accurate for sparse network topologies. The work in [5] developed a Markov chain with state-dependent arrival rates to model call blocking in arbitrary mesh topologies and fixed routing; it was extended in [6] to alternate routing. A more tractable model was presented in [7] to recursively compute blocking probabilities assuming that the load on link  $i$  of a path depends only on the load of link  $i - 1$ . Finally, a study of call blocking under non-Poisson input traffic was presented in [8], under the assumption that link loads are statistically independent.

Other wavelength allocation schemes, as well as dynamic routing are harder to analyze. First-fit wavelength allocation was studied using simulation in [4], and it was shown to perform better than random allocation, while an analytical overflow model for first-fit allocation was developed in [9]. A dynamic routing algorithm that selects the least loaded path-wavelength pair was also studied in [9], and in [10] an unconstrained dynamic routing scheme with a number of wavelength allocation policies was evaluated. Except in [7,11], all other studies assume that either all or none of the wavelength routers have wavelength conversion capabilities. The work in [7] takes a probabilistic approach in modeling wavelength conversion by introducing the converter density, which represents the probability that a node is capable of conversion independently of other nodes in the network. A dynamic programming algorithm to determine the location of converters on a single path that minimizes average or maximum blocking probability was developed in [11] under the assumption of independent link loads.

Most of the approximate analytical techniques developed for computing blocking probabilities in wavelength routing networks [4,5,6,8,9,10,11] amount to the well-known *link decomposition* approach [12], while the development of some techniques is based on the additional assumption that link loads are also independent. Link decomposition has been extensively used in conventional circuit-



switched networks where there is no requirement for the *same* wavelength to be used on successive links of the path taken by a call. The accuracy of these underlying approximations also depends on the traffic load, the network topology, and the routing and wavelength allocation schemes employed. While link decomposition techniques make it possible to study the qualitative behavior of wavelength routing networks, more accurate analytical tools are needed to evaluate the performance of these networks efficiently, as well as to tackle complex network design problems, such as selecting the nodes where to employ converters.

We have developed an iterative *path decomposition* algorithm [13] for analyzing arbitrary network topologies. Specifically, we analyze a given network by decomposing it into a number of path sub-systems. These sub-systems are analyzed in isolation using our approximation algorithm for computing blocking probabilities in a single path of a network [14]. The individual solutions are appropriately combined to form a solution for the overall network, and the process repeats until the blocking probabilities converge. Our approach accounts for the correlation of both link loads and link blocking events, giving accurate results for a wide range of loads and network topologies where only a fixed but arbitrary subset of nodes are capable of wavelength conversion. Therefore, our algorithm can be an important tool in the development and evaluation of converter placement strategies. Also, in [15] we studied the first-fit and most-used wavelength allocation policies, and we showed that they have almost identical performance in terms of blocking probability for all calls in the network. We also demonstrated that the blocking probabilities under the random wavelength allocation policy with no converters and with converters at all nodes provide upper and lower bounds for the values of the blocking probabilities under the first-fit and most-used policies.

All previous studies of call blocking probabilities have considered single-class wavelength routing networks. However, future networks will be utilized by a wide range of applications with varying characteristics in terms of their arrival rates and call holding times. In this paper, we present a method to extend the results in [14] and [13] to multi-class optical networks. To the best of our knowledge, this is the first time that multi-class wavelength routing networks are analyzed.

The development of our approximate techniques involves two steps. The arrival process of calls on some routes is first modified slightly to obtain a *modified multi-class network model*. Next, all classes of calls on a particular route are aggregated to give an *equivalent single-class model*. This equivalent model has the same call blocking probability on any given route as the modified multi-class network, and can be solved using our previous algorithms.

In Section 2, we describe the network under study. In Section 3, we explain how the modified multi-class model and the equivalent single-class model are obtained for a single path of a network. In Section 4, we describe a decomposition algorithm for mesh networks. Section 5 presents numerical results, and we conclude the paper in Section 6.

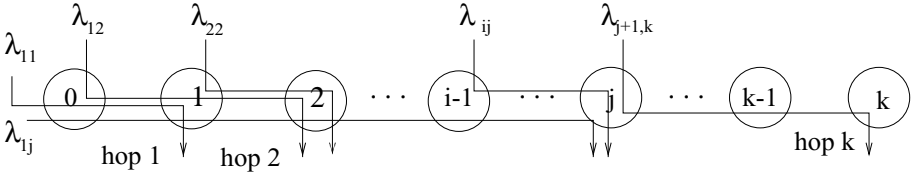


Fig. 1. A  $k$ -hop path

## 2 The Multi-class Wavelength Routing Network

We consider a wavelength routing network with an arbitrary topology. Each link in the network supports exactly  $W$  wavelengths, and each node is capable of transmitting and receiving on any of these  $W$  wavelengths. Call requests between a source and a destination node arrive at the source according to a Poisson process with a rate that depends on the source-destination pair. If the request can be satisfied, an optical circuit is established between the source and destination for the duration of the call. Further, calls between any two nodes may be of several classes. Without any loss of generality, we may assume that there are  $R$  classes of calls. Call holding times are assumed to be exponentially distributed, with a mean that depends on the class of the call.

In our model, we allow some of the nodes in the network to employ wavelength converters. These nodes can switch an incoming wavelength to an arbitrary outgoing wavelength. (When there are converters at all nodes, the situation is identical to that in classical circuit-switching networks, a special case of the more general scenario discussed here.) If no wavelength converters are employed in the path between the source and the destination, a call can only be established if the *same* wavelength is free on all the links used by the call. If there exists no such wavelength, the call is blocked. This is known as the *wavelength continuity* requirement (see [13,14]), and it increases the probability of call blocking. On the other hand, if a call can be accommodated, it is randomly assigned one of the wavelengths that are available on the links used by the call<sup>1</sup>. Thus, we only consider the random wavelength assignment policy in this paper.

Since many of our results are developed in terms of a single path in a wavelength routing network, we introduce some relevant notation. A  $k$ -hop path (see Figure 1) consists of  $k + 1$  nodes. Nodes  $(i - 1)$  and  $i$ ,  $1 \leq i \leq k$ , are said to be connected by link (hop)  $i$ . Calls originating at node  $i - 1$  and terminating at node  $j$  use hops  $i$  through  $j$ ,  $j \geq i \geq 1$ , which we shall denote by the pair  $(i, j)$ . Calls between these two nodes may belong to one of  $R$  classes, and these calls are said to use *route*  $(i, j)$ . We also define the following parameters.

- $\lambda_{ij}^{(r)}$ ,  $j \geq i$ ,  $1 \leq r \leq R$ , is the Poisson arrival rate of calls of class  $r$  that originate at node  $(i - 1)$  and terminate at node  $j$ .

<sup>1</sup> In a path with wavelength converters, a wavelength is randomly assigned within each segment of the path whose starting and ending nodes are equipped with converters.

- $1/\mu_{ij}^{(r)}$  is the mean of the exponentially distributed service time of calls of class  $r$  that originate at node  $(i-1)$  and terminate at node  $j$ . We also let  $\rho_{ij}^{(r)} = \lambda_{ij}^{(r)} / \mu_{ij}^{(r)}$ .
- $N_{ij}^{(r)}(t)$  is the number of active calls at time  $t$  on segment  $(i, j)$  belonging to class  $r$ .
- $F_{ij}(t)$  is the number of wavelengths that are free on all hops of segment  $(i, j)$  at time  $t$ . A call that arrives at time  $t$  and uses route  $(i, j)$  is blocked if  $F_{ij}(t) = 0$ .

### 3 Blocking Probabilities in a Single Path of a Network

#### 3.1 The Single-Class Case

In this section we briefly review some of our previous results for a path of a single-class wavelength routing network. Consider the  $k$ -hop path shown in Figure 1. Let the state of this system at time  $t$  be described by the  $k^2$ -dimensional process:

$$\underline{X}_k(t) = (N_{11}(t), N_{12}(t), \dots, N_{kk}(t), F_{12}(t), \dots, F_{1k}(t), F_{23}(t), \dots, F_{(k-1)k}(t)) \quad (1)$$

A closer examination of the process  $\underline{X}_k(t)$  reveals that it is not time-reversible (see [14]). This result is true in general, when  $k \geq 2$  and  $W \geq 2$ .

Since the number of states of process  $\underline{X}_k(t)$  grows very fast with the number  $k$  of hops in the path and the number  $W$  of wavelengths, it is not possible to obtain the call blocking probabilities directly from the above process. Consequently, an approximate model was constructed in [14] to analyze a single-class,  $k$ -hop path of a wavelength routing network. The approximation consists of modifying the call arrival process to obtain a time-reversible Markov process that has a closed-form solution. To illustrate our approach, let us consider the Markov process corresponding to a 2-hop path:

$$\underline{X}_2(t) = (N_{11}(t), N_{12}(t), N_{22}(t), F_{12}(t)) \quad (2)$$

We now modify the arrival process of calls that use both hops (a Poisson process with rate  $\lambda_{12}$  in the exact model) to a state-dependent Poisson process with rate  $\Lambda_{12}$  given by:

$$\Lambda_{12}(n_{11}, n_{12}, n_{22}, f_{12}) = \lambda_{12} \frac{f_{12}(W - n_{12})}{f_{11}f_{12}} \quad (3)$$

The arrival process of other calls remain as in the original model. As a result, we obtain a new Markov process  $\underline{X}'_2(t)$  with the same state space and the same state transitions as process  $\underline{X}_2(t)$ , but which differs from the latter in some of the state transition rates.

We made the observation in [14] that under the new arrival process (3) for calls using both hops, the Markov process  $\underline{X}'_2(t)$  is time-reversible and the stationary vector  $\pi$  is given by:

$$\pi(n_{11}, n_{12}, n_{22}, f_{12}) = \frac{1}{G_2(W)} \frac{\rho_{11}^{n_{11}} \rho_{12}^{n_{12}} \rho_{22}^{n_{22}}}{n_{11}! n_{12}! n_{22}!} \times \frac{\binom{f_{11}}{f_{12}} \binom{n_{11}}{W - n_{12} - n_{22} - f_{12}}}{\binom{W - n_{12}}{W - n_{12} - n_{22}}} \quad (4)$$

where  $G_k(W)$  is the normalizing constant for a  $k$ -hop path with  $W$  wavelengths.

Let  $P(n_{11}, n_{12}, n_{22})$  be the marginal distribution over the states for which  $N_{ij}(t) = n_{ij}$ ,  $1 \leq i \leq j \leq 2$ . It can be verified [14] that

$$P(n_{11}, n_{12}, n_{22}) = \frac{1}{G_2(W)} \frac{\rho_{11}^{n_{11}}}{n_{11}!} \frac{\rho_{12}^{n_{12}}}{n_{12}!} \frac{\rho_{22}^{n_{22}}}{n_{22}!} \quad (5)$$

Likewise, for a  $k$ -hop path,  $k \geq 2$ , with the modified state-dependent Poisson arrival process, the marginal distribution over the states for which  $N_{ij}(t) = n_{ij}$ ,  $1 \leq i \leq j \leq k$ , is given by:

$$P(n_{11}, n_{12}, \dots, n_{kk}) = \frac{1}{G_k(W)} \prod_{\{(i,j)|1 \leq i \leq j \leq k\}} \frac{\rho_{ij}^{n_{ij}}}{n_{ij}!} \quad (6)$$

It is easily seen that this distribution is the same as in the case of a network with wavelength converters at each node. An interesting feature of having wavelength converters at every node is that the network has a product-form solution even when there are multiple classes of calls on each route, as long as call arrivals are Poisson, and holding times are exponential [16,12]. Further, when calls of all classes occupy the same number of wavelengths, we can aggregate classes to get an equivalent single class model with the same steady-state probability distribution over the aggregated states, as we show next.

### 3.2 The Multi-class Case

Let us now consider a  $k$ -hop path with wavelength converters at all nodes, and with  $R$  classes of calls. If  $\lambda_{ij}^{(r)}$ ,  $1 \leq i \leq j \leq k$ ,  $1 \leq r \leq R$ , is the arrival rate of calls of class  $r$  on route  $(i, j)$ , and  $1/\mu_{ij}^{(r)}$ ,  $1 \leq i \leq j \leq k$ ,  $1 \leq r \leq R$ , is the mean of the exponential holding time of calls of class  $r$ , the probability of being in state  $\underline{n} = (n_{11}^{(1)}, n_{11}^{(2)}, \dots, n_{11}^{(R)}, n_{12}^{(1)}, \dots, \dots, n_{kk}^{(R)})$  is given by:

$$P(\underline{n}) = \frac{1}{G_k(W)} \left( \prod_{\{(i,j)|1 \leq i \leq j \leq k\}} \prod_{r=1}^R \frac{(\rho_{ij}^{(r)})^{n_{ij}^{(r)}}}{n_{ij}^{(r)}!} \right) \quad (7)$$

Let  $\sigma_{ij} = \sum_r \rho_{ij}^{(r)}$  and  $s_{ij} = \sum_r n_{ij}^{(r)}$ . As defined,  $s_{ij}$  is the total number of calls of all classes that use segment  $(i, j)$  of the path, and  $\sigma_{ij}$  is the total offered load of these calls. Taking the summation of (7) over all states such that  $\sum_r n_{ij}^{(r)} = s_{ij}$ ,  $1 \leq i \leq j \leq k$ , we obtain:

$$P'(s_{11}, s_{12}, \dots, s_{kk}) = \sum_{\{\underline{n} | \sum_r n_{ij}^{(r)} = s_{ij}\}} P(\underline{n}) = \frac{1}{G_k(W)} \prod_{\{(i,j)|1 \leq i \leq j \leq k\}} \frac{\sigma_{ij}^{s_{ij}}}{s_{ij}!} \quad (8)$$

Observe that this is identical to the solution (6) for the single-class case obtained by substituting  $\sigma_{ij}$  by  $\rho_{ij}$  and  $s_{ij}$  by  $n_{ij}$  in (8).

Based on the above results, we conclude that by employing class aggregation on a multi-class path with converters at all nodes, we obtain a system equivalent to a single-class path with converters. In Section 3.1, we showed that the modified single-class wavelength routing network without converters has a steady-state marginal distribution similar to the exact single-class network with converters. We now show that a modified multi-class network without wavelength converters can also be subjected to class aggregation that results in an equivalent single-class model. The modification applied to the arrival process of calls is similar to the single-class case, and it is given by:

$$A_{ij}^{(r)}(\underline{x}) = \lambda_{ij}^{(r)} \frac{f_{ij} \left( \sum_{l=1}^i s_{li} + f_{ii} \right) \left( \sum_{l=1}^{i+1} s_{l(i+1)} + f_{(i+1)(i+1)} \right) \cdots \left( \sum_{l=1}^{j-1} s_{l(j-1)} + f_{(j-1)(j-1)} \right)}{f_{ii} f_{(i+1)(i+1)} \cdots f_{jj}} \quad (9)$$

Then, the probability that the equivalent single-class network without converters is in state  $S_{\underline{x}} = (s_{11}, s_{12}, \dots, s_{1k}, s_{22}, \dots, s_{kk}, f_{12}, f_{13}, \dots, f_{(k-1)k})$  is given by:

$$\pi(S_{\underline{x}}) = \left( \prod_{i,j} \frac{s_{ij}!}{\sigma_{ij}^{s_{ij}}} \right) \left( \prod_{l=2}^k \frac{\binom{f_{1(l-1)}}{f_{1l}} \left\{ \prod_{m=2}^{l-1} \binom{f_{m(l-1)} - f_{(m-1)(l-1)}}{f_{ml} - f_{(m-1)l}} \right\} \binom{f_{ll} + n_{ll} - f_{(l-1)(l-1)}}{f_{ll} - f_{(l-1)l}}}{\binom{f_{ll} + n_{ll}}{f_{ll}}} \right) \quad (10)$$

Once again, the parameters of the single-class model are given by:

$$s_{ij} = \sum_{r=1}^R n_{ij}^{(r)} \quad 1 \leq i < j \leq k, \quad \sigma_{ij} = \sum_{r=1}^R \rho_{ij}^{(r)} \quad 1 \leq i < j \leq k \quad (11)$$

### 3.3 Blocking Probabilities in the Multi-class Case

Since the arrival rate of calls of each class on each route is Poisson, the blocking probability,  $Q_{ij}^{(r)}$ , of a call of class  $r$  using route  $(i, j)$  is just the fraction of time that there are no wavelengths that are free on all hops along route  $(i, j)$  (see the *PASTA* theorem in [17]). Thus, we have:

$$Q_{ij}^{(r)} = \lim_{\tau \rightarrow \infty} \frac{\int_{t=0}^{\tau} I_{\{F_{ij}(t)=0\}} dt}{\tau}, \quad \text{where} \quad I_{\{F_{ij}(\tau)=0\}} = \begin{cases} 1, & \text{if } F_{ij}(\tau) = 0 \\ 0, & \text{otherwise} \end{cases} \quad (12)$$

As can be seen, the blocking probability is class-independent.

Next, we focus on the call blocking probabilities in the modified model. The arrival process of calls of class  $r$  on route  $(i, j)$  is a state-dependent Poisson process whose rate at time  $\tau$ ,  $A_{ij}^{(r)}(\tau)$  is a function of the state  $\underline{X}(\tau)$  of the process, and is given by:

$$A_{ij}^{(r)}(\tau) A_{ij}^{(r)}(\underline{X}(\tau)) = \lambda_{ij} \frac{F_{ij}(\tau) \prod_{k=i}^{j-1} \left( \sum_{l=1}^k N_{lk}(\tau) + F_{kk}(\tau) \right)}{F_{ii} F_{i+1, i+1} \cdots F_{jj}} \quad (13)$$

Note that the modified arrival process satisfies the criterion:

$$\frac{\Lambda_{ij}^{(r_1)}(\underline{x})}{\Lambda_{ij}^{(r_2)}(\underline{x})} = \frac{\lambda_{ij}^{(r_1)}}{\lambda_{ij}^{(r_2)}} \quad 1 \leq r_1, r_2 \leq r \quad (14)$$

By applying the *PASTA* theorem conditioned on being in state  $\underline{x}$ , the conditional call blocking probability,  $\mathcal{P}_{ij}^{(r)}(\underline{x})$ , of calls of class  $r$  on route  $(i, j)$  is given by the fraction of time spent in state  $\underline{x}$  in which there are no wavelengths that are free on all hops of route  $(i, j)$ . Therefore:

$$\mathcal{P}_{ij}^{(r)}(\underline{x}) = \lim_{\tau \rightarrow \infty} \frac{\int_{t=0}^{\tau} I_{\{F_{ij}(t)=0, \underline{X}(t)=\underline{x}\}} dt}{\int_{t=0}^{\tau} I_{\{\underline{X}(t)=\underline{x}\}} dt} = \begin{cases} 1, & \text{if } f_{ij} = 0 \\ 0, & \text{otherwise} \end{cases} \quad (15)$$

Let  $P_{ij}^{(r)}$  be the unconditional probability that a call of class  $r$ , on route  $(i, j)$  gets blocked in the modified multi-class model. This is given by:

$$P_{ij}^{(r)} = \frac{\sum_{\underline{x}} \Lambda_{ij}^{(r)}(\underline{x}) \pi(\underline{x}) \mathcal{P}_{ij}^{(r)}(\underline{x})}{\sum_{\underline{x}} \Lambda_{ij}^{(r)}(\underline{x}) \pi(\underline{x})} = \frac{\sum_{\{\underline{x} | f_{ij}=0\}} \Lambda_{ij}^{(r)}(\underline{x}) \pi(\underline{x})}{\sum_{\underline{x}} \Lambda_{ij}^{(r)}(\underline{x}) \pi(\underline{x})} \quad (16)$$

and can also be seen to be independent of the class  $r$ . Thus, by computing the blocking probability on the equivalent single-class path, we can obtain the solution to the multi-class path.

## 4 Blocking Probabilities in Mesh Topologies

The solution to single-class networks with wavelength converters at an arbitrary subset of nodes has been presented in [14,13]. This solution involves decomposition of the network into short path segments with two or three hops, and analyzing these approximately using expression (4). The solutions to individual segments are appropriately combined to obtain a value for the blocking probability of calls that traverse more than one segment. The effect of the wavelength continuity requirement is captured by an approximate *continuity factor* that is used to increase the blocking probability of calls continuing to the next segment to account for the possible lack of common free wavelengths in the two segments. The process repeats until the blocking probabilities converge. By applying the transformations in (11), the same algorithms may be used to calculate blocking probabilities for multi-class networks. Specifically, we use these steps for a network with  $R$  classes of calls:

1. **Path decomposition:** Decompose the multi-class mesh network topology into  $L$  single-path sub-systems using the algorithm in [13].
2. **Time-reversible process approximation:** For each single-path sub-system, modify the arrival process as given by expression (9) to obtain an approximate time-reversible Markov process for the path.

3. **Class aggregation:** For each sub-system, apply the transformations in (11) to obtain an equivalent single-class path sub-system.
4. **Calculation of blocking probabilities:** For each path sub-system, obtain the blocking probabilities as follows. If the path is at most three hops long, use expressions (16) and (10) directly. If the path sub-system is longer than three hops, analyze it by decomposing it into 2- or 3-hop paths which are solved in isolation, and combine the individual solutions to obtain the blocking probabilities along the original longer path (see [14]).
5. **Convergence:** Repeat Steps 2 to 4, after appropriately modifying the original arrival rates to each single-path sub-system to account for the new values of the blocking probabilities obtained in Step 4 (see [13]), until the blocking probabilities converge within a certain tolerance.

## 5 Numerical Results

In this section, we validate the approximate method described in Section 4 by comparing the blocking probabilities for each route as obtained from the approximate method with those obtained through simulation of the exact model.

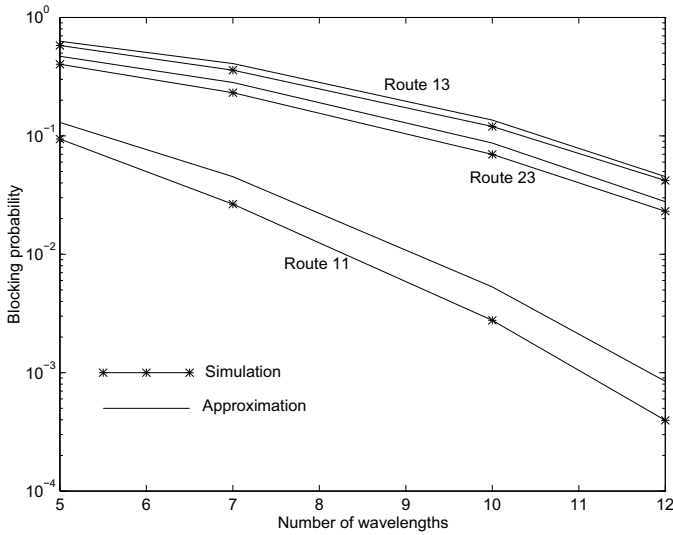
We first provide results for 3-hop paths which are the basic blocks of our decomposition algorithm (results for 2-hop paths can be found in [18]). In Table 1 we show the arrival and service rates for calls on each route  $(i, j)$ ,  $1 \leq j \leq 3$ , of a 3-hop path. There are  $R = 3$  classes of calls for each route. In Figure 2, we plot the blocking probability against the number  $W$  of wavelengths for three (out of the six) types of calls in this path: calls using Route  $(1, 1)$ , i.e., the first hop of the path, calls on Route  $(2, 3)$ , that is, those using the last two hops, and calls on Route  $(1, 3)$  using all three hops of the path. As we can see, the blocking probability decreases as  $W$  increases, as expected. We also observe that calls on Route  $(1, 3)$  (i.e., calls using all three hops of the path) experience the highest blocking probability, again as expected. Most importantly, however, we can see that there is good agreement between the values of the blocking probabilities obtained through our analytical technique and those obtained through simulation. Similar results have been obtained for different values for the arrival and service parameters and for different number of classes, indicating that our approximate method is accurate over a wide range of network characteristics.

Next, we consider a network with topology similar to the NSF network, shown in Figure 3. There are 16 nodes, and 240 uni-directional routes. There are three classes of calls on each route. The arrival and service rates of calls of a particular class are the same on each route, and are shown in Table 2. The blocking probabilities are plotted in Figure 4 for four routes, as a function of the number of wavelengths on each link. Route A is a single-hop route from nodes 1 to 5. Route B has two hops, connecting node 1 to node 3 via node 2. Route C has three hops, connecting node 1 to node 4 via nodes 2 and 3. Route D has four hops, connecting node 4 to node 5 via nodes 3, 2 and 1.

From Figure 4 we can see that the length of the path used by a call considerably affects the blocking probability experienced by the call, an observation that

**Table 1.** Arrival and service parameters for a 3-hop path

Route ( $i, j$ )	Class 1		Class 2		Class 3	
	$\lambda$	$\mu$	$\lambda$	$\mu$	$\lambda$	$\mu$
(1,1)	3.0	3.0	1.0	3.0	1.0	3.0
(1,2)	1.0	6.0	2.0	6.0	1.0	6.0
(1,3)	1.0	2.0	1.0	2.0	2.0	2.0
(2,2)	1.0	3.0	1.0	3.0	2.0	3.0
(2,3)	1.0	2.0	1.0	2.0	3.0	2.0
(3,3)	3.0	6.0	1.0	6.0	1.0	6.0



**Fig. 2.** Blocking probabilities for a 3-hop path and the parameters shown in Table 1

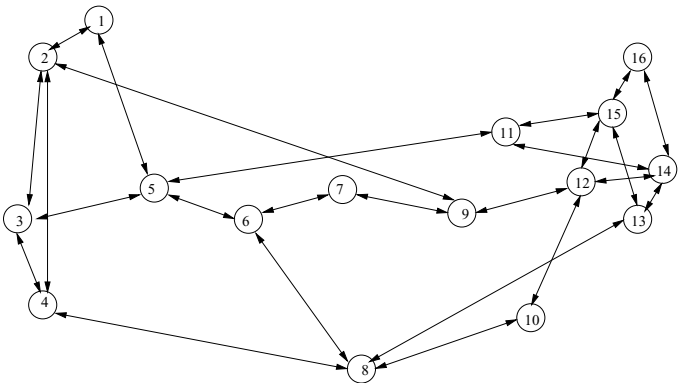
is consistent with all our previous results in this section. Specifically, for a given number  $W$  of wavelengths, the blocking probability increases with the number of hops in a route, such that calls on Route A (a single-hop path) have the lowest blocking probability while calls on Route D (a four-hop path) the highest. Further, the results indicate that our approximation method can be used to estimate accurately the blocking probabilities for all calls in the network.

Results similar to the ones presented in Figures 2,4 have been obtained for a wide range of traffic loads and different classes of calls, and for other network topologies (see [18]). Our main conclusion is that our approximate analytical technique can be applied to compute the call blocking probabilities in wavelength routing networks of realistic size and topology. The approximate technique affords a significant reduction in the time for computation of blocking probabilities. For instance, the simulation program took approximately 100 minutes while running on a Sun-sparc Ultra 10 workstation, to compute call blocking

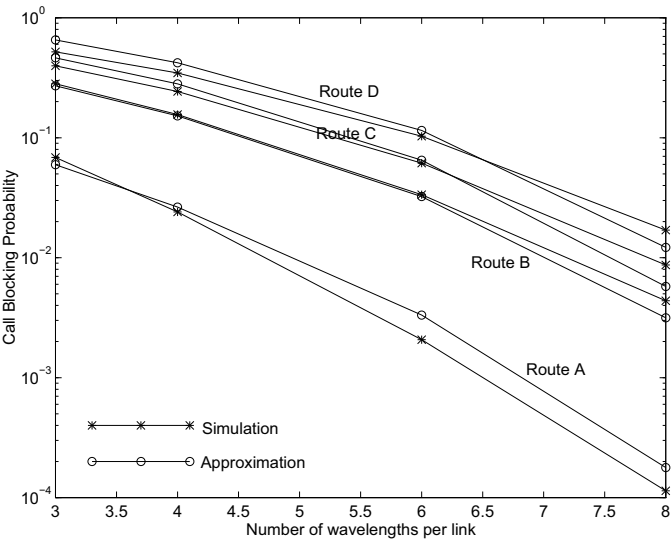


**Table 2.** Arrival and service parameters for the NSF Network

	Class 1	Class 2	Class 3
$\lambda$	0.1	0.2	0.15
$\mu$	3.0	6.0	4.0



**Fig. 3.** The NSF network



**Fig. 4.** Blocking probabilities for the NSF network

probabilities for the network with topology similar to the NSF network. The approximation method took less than a minute for the same network. In addition, the simulation program runs until there are at least 100000 call arrivals of every class. For call blocking probabilities less than  $10^{-4}$ , the simulation program must consider greater than  $10^6$  call arrivals of each class, resulting in an even longer computation time.

## 6 Concluding Remarks

We have considered the problem of computing call blocking probabilities in multi-class wavelength routing networks which employ the random wavelength allocation policy. Our approach consists of modifying the call arrival process to obtain an approximate multi-class network model, using class aggregation to map this to an equivalent single-class network, and employing path decomposition algorithms on the latter to determine the call blocking probabilities.

## References

1. A. Hill, A. Salek, and K. Sato (Eds.). Special issue on high-capacity optical transport networks. *IEEE JSAC*, 16(7), Sep. 1998.
2. B. Ramamurty and B. Mukherjee. Wavelength conversion in WDM networking. *IEEE JSAC*, 16(7):1061–1073, Sep. 1998.
3. R. Barry and P. Humblet. Models of blocking probability in all-optical networks with and without wavelength changers. *IEEE JSAC*, 14(5):858–867, June 1996.
4. M. Kovacevic and A. Acampora. Benefits of wavelength translation in all-optical clear-channel networks. *IEEE JSAC*, 14(5):868–880, June 1996.
5. A. Birman. Computing approximate blocking probabilities for a class of all-optical networks. *IEEE JSAC*, 14(5):852–857, June 1996.
6. H. Harai, M. Murata, and H. Miyahara. Performance of alternate routing methods in all-optical switching networks. *Proc. INFOCOM '97*, pp. 517–525, April 1997.
7. S. Subramaniam, M. Azizoglu, and A. Somani. All-optical networks with sparse wavelength conversion. *IEEE/ACM Trans. Network.*, 4(4):544–557, Aug. 1996.
8. S. Subramanian *et al.* A performance model for wavelength conversion with non-poisson traffic. *Proc. INFOCOM '97*, pp. 500–507, April 1997.
9. E. Karasan and E. Ayanoglu. Effects of wavelength routing and selection algorithms on wavelength conversion gain in WDM optical networks. *IEEE/ACM Trans. Network.*, 6(2):186–196, April 1998.
10. A. Mokhtar and M. Azizoglu. Adaptive wavelength routing in all-optical networks. *IEEE/ACM Trans. Network.*, 6(2):197–206, April 1998.
11. S. Subramaniam, M. Azizoglu, and A. K. Somani. On the optimal placement of wavelength converters in wavelength-routed networks. *Proc. INFOCOM '98*, pp. 902–909, April 1998.
12. A. Girard. *Routing and Dimensioning in Circuit-Switched Networks*. Addison Wesley, Reading, MA, 1990.
13. Y. Zhu, G. N. Rouskas, and H. G. Perros. Blocking in wavelength routing networks, Part II: Mesh topologies. *Proc. ITC 16*, pp. 1321–1330, June 1999.
14. Y. Zhu, G. N. Rouskas, and H. G. Perros. Blocking in wavelength routing networks, Part I: The single path case. *Proc. INFOCOM '99*, pp. 321–328, March 1999.

15. Y. Zhu, G. N. Rouskas, and H. G. Perros. Bounds on the blocking performance of allocation policies in wavelength routing networks and a study of the effects of converters. TR-99-01, NCSU, Jan. 1999.
16. F. P. Kelly. *Reversibility and Stochastic Networks*. John Wiley & Sons, NY, 1979.
17. R. W. Wolff. Poisson arrivals see time averages. *Oper. Res.*, 30(2):223–231, 1982.
18. S. Ramesh, G. N. Rouskas, and H. G. Perros. Computing call blocking probabilities in multi-class wavelength routing networks. TR-99-08, NCSU, May 1999.

# A Comparative Study of Mesh and Multi-ring Designs for Survivable WDM Networks

Thanyaporn Iamvasant, Charoenchai Baworntummarat,  
and Lunchakorn Wuttisittikulkij

Department of Electrical Engineering  
Faculty of Engineering  
Chulalongkorn University  
Bangkok, Thailand 10330  
Tel. +662 2186512 Fax. +662 2518991  
[lunch@ee.eng.chula.ac.th](mailto:lunch@ee.eng.chula.ac.th)

**Abstract.** In this paper, two distinct optical network design approaches, namely mesh and multi-ring, for survivable WDM networks are investigated. The main objective is to compare these two design approaches in terms of network costs so that their merits in practical environments can be identified. In the mesh network design, a new mathematical model based on integer liner programming (ILP) and a heuristic algorithm are presented for achieving a minimal cost network design. In the multi-ring network design, a heuristic algorithm that can be applied to large network problems is proposed. The influence of wavelength conversion and the number of wavelengths multiplexed in a fiber on system designs are also discussed. Based on the simulation results, the redundancy quantities required for full protection in multi-ring approach are significantly larger in comparison to the minimal cost mesh counterpart.

## 1 Introduction

Recently, wavelength division multiplexing (WDM) has been seen as a promising technology for realizing future broadband networks to support the increasing bandwidth demands of various emerging applications, such as multimedia and web-browsing. In such networks, a few number of wavelength channels can be multiplexed into a single fiber, each operating at a few Gbit/s. Therefore, these types of networks are expected to offer an aggregate capacity in the order of Tbit/s, serving as a viable technology to overlay the existing transport networks.

One of the key issues associated with WDM network designs is the problem of wavelength allocation. Over the past few years, many research activities have made considerable efforts to solve this problem [1,2,3,4]. Moreover, some of these studies also include the network protection issue into their design considerations. This is because there is an increasing concern on the impacts of network failures in modern communication systems. It is important that certain network protection measures must be provided at the network design and dimensioning stage.

This paper studies the problem of network resource allocation in WDM networks employing wavelength routing technique. The key objective is to determine how fiber and wavelength resources can be simultaneously assigned to satisfy traffic demands

while providing full protection against all single link failures. The solution techniques for this problem are based on two design approaches: mesh and multi-ring.

In the mesh design, two path restoration schemes, namely a minimal cost protection and a single link basis protection, are examined. For the minimal cost protection approach [5], in the events of failures all optical connections are subject to be rearranged even when they may not be directly interrupted by the failures. Accordingly, this particular approach can allow, in principle, the design to be very efficient and result in the minimal network cost. This protection technique is therefore called the minimal cost approach (MC). For the restoration on a single link basis (SLB), only interrupted traffic connections are rerouted and other connections remain unchanged. Failures on different links along an active path can have different restoration paths depending on the place of link failure. Consequently, this approach is called the single link basis approach (SLB).

In order to illustrate the difference between the MC and SLB approaches, an example of a network scenario is given in Figure 1. In this example, the connections (1,6) and (3,4) are set up along the physical routes of 1-4-6 and 3-2-4 respectively under normal operation. Now, consider a situation when a failure occurs on the link between nodes 1 and 4. In SLB approach, only the connection (1,6) is subject to reroute as it is directly affected by the failure and in this example the new route chosen is through nodes 1-2-3-6 resulting in an extra wavelength channel being needed on link 2-3. In contrast, the MC approach requires no additional wavelength as both connections are re-arranged.

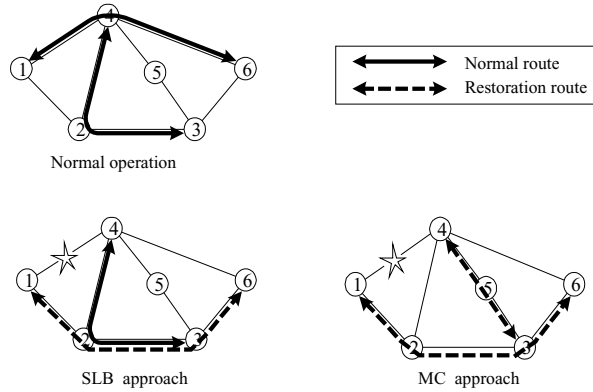


Fig. 1. An example illustrating the difference between the MC and SLB approaches

Although the minimal cost protection technique is very cost-efficient, it may not be suitable for practical use, due to unnecessary reconfigurations of the entire optical nodes in the networks. However, the design outcomes of this protection technique are useful for comparisons in terms of network costs with the other path restoration schemes such as SLB approach.

In the ring design, realizing a large-scale network using a single ring may not be practical due to the low efficiency of network bandwidth utilization. A new design technique using multiple rings described in [6] has been proposed. Instead of using a large single ring to cover the entire network, the network is divided into a number of

rings. These rings are used to support all traffic of the network. The criterion used in the design is that the traffic demands between each node pair must be accommodated over a single ring. No traffic is allowed to get across between rings in order to avoid complicated control and management. Consequently, the main problem of this design is how to select the rings from the number of possible rings in network in order to minimize the number of required fibers.

The multi-ring design may not be as efficient as the mesh design in terms of network utilization but it is an interesting alternative and currently attracts much attention. Firstly, since its structure is much simpler than that of the mesh, only add-drop multiplexers are needed whereas optical cross-connects would be required for mesh. Secondly, each unit of rings in the network can operate independently, thus the control and management can be fully distributed. Thirdly, the multi-ring design provides a full network protection against certain types of failures: all single-link and all single-node failures. This restoration feature is particularly important in future optical network which requires high level of network protection. Because of the multi-ring's simple structure, the entire restoration process in the ring network can be achieved through automatic hardware reconfiguration without rerouting the entire path, hence it is very fast and reliable. However a disadvantage of the multi-ring approach is that protection cost is 100% of unprotected rings.

The key issue that is addressed in this paper is determining and comparing the network costs between the two different design schemes in a quantitative manner. In addition, this paper also considers two different systems, namely wavelength path (WP) and virtual wavelength path (VWP). The VWP system differs from the WP system in that its node configurations include an additional wavelength conversion capability. Therefore, when setting up an optical path for a connection, the VWP can assign wavelengths on a link-by-link basis, whereas the WP must choose a single wavelength for all links along the entire physical route.

## 2 Minimal Cost Protection Technique

### 2.1 Integer Linear Programming (ILP)

Given a set of traffic demands and a physical network topology, the network topology is modeled as an undirected graph  $G(v, \epsilon)$  where  $v$  is a set of nodes with size  $N$  and  $\epsilon$  is a set of edges with size  $L$ . Let  $M$  be the number of wavelengths multiplexed into a fiber. Let  $X_i$  be the number of fibers (or capacity) on  $i^{\text{th}}$  edge and the characteristics of each fiber is assumed to be bi-directional and can only contain one wavelength ( $M=1$ ), the objective function of this model is

$$\text{Min} : \sum X_i \quad \text{where } i \in \epsilon \quad (1)$$

The constraint to formulate the equation is derived from the minimum flow on a cut set which is defined as follows. The network graph is partitioned into two parts,  $G_1$  and  $G_2$  with cut set  $\epsilon_c$  which corresponds to a set of edges that has one endpoint in  $G_1$  and the other in  $G_2$ . The possible minimum traffic of a cut set,  $d_{\min}(c)$ , is the summation of each traffic demand which has a source and a destination lying on

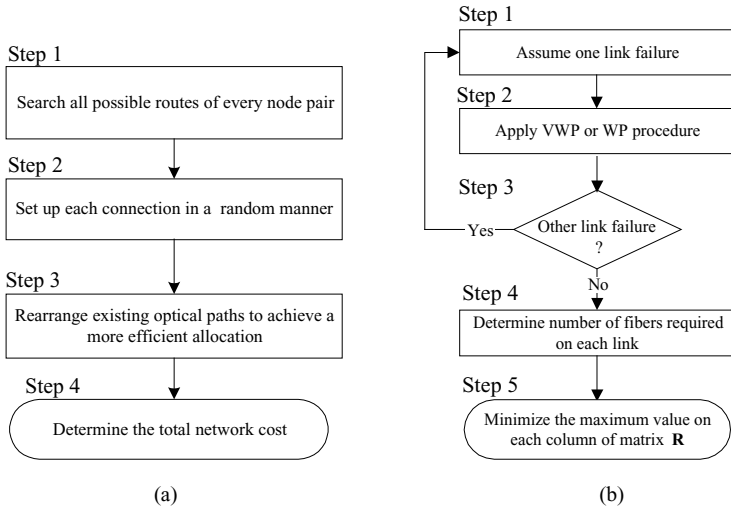
different parts. As a result, the capacity of any cut set must be larger than or equal to the possible minimum traffic:

$$\sum X_i \geq d_{\min}(c) \quad \text{where } i \in \varepsilon_c \quad (2)$$

Now, we can determine the network capacity from the linear formulations related to every possible cut set of the network graph. This capacity only provides for the traffic demands in normal operation. In order to extend the model to cover the protection requirements against all single link failures, additional constraints are included as follows. An edge is removed from the network graph in turn, simulating each link failure event. Given a network graph with an edge removed, the same formulation of the minimum flow on a cut set is applied, producing a new set of required constraints. By applying this procedure to all edges, a total of  $L$  additional new sets of constraints would result, covering all link failure scenarios. All these constraints are sufficient to determine the minimal network cost with full protection. However, this model does not provide an exact solution, instead it can only be used as a lower bound on the network cost required.

## 2.2 Heuristic Algorithm

Due to the complexity of the problem, the design process is divided into three procedures: path accommodation, wavelength assignment, and restoration. The wavelength assignment procedure used in this paper is adopted from [7]. Since the wavelength assignment in VWP system is considered on a link-by-link basis, only the path accommodation procedure is required. Firstly, we will explain the path accommodation procedure. The flow chart of path accommodation is shown in Figure 2 (a). The outline is described as follows:



**Fig. 2.** Procedure of (a) the path accommodation and (b) MC restoration

Step 1: All possible routes between each node pair are searched  
 Step 2: Randomly select an appropriate physical route for each traffic connection  
 Step 3: After all connections have been established, the next step is an attempt to rearrange the existing allocation, so that the overall network cost is reduced. To accomplish this, each connection is reconsidered in turn to see whether they can be redirected on another route that leads to more efficient wavelength resource utilization  
 Step 4: As the VWP system assigns wavelengths on a link-by-link basis, the number of wavelength channels required on each link is equal to the number of optical connections passing through the corresponding link. The number of fibers required on each link can be obtained from  $\lceil P/M \rceil$  where  $P$  is the number of optical paths going through the link and  $M$  is the number of wavelengths multiplexed in one fiber. Note that  $\lceil X \rceil$  is the smallest integer greater than or equal to  $X$ . Consequently, the network cost can be obtained from the summation of the number of required fibers of every link times the number of wavelengths. In WP system, the wavelength assignment procedure is applied before determining the protection cost

The following restoration algorithm guarantees 100% protection against any single link failure in the network and it can be applied to both VWP and WP systems. The outline of this algorithm is shown in Figure 2 (b).

Step 1: A fiber link  $i$  is removed from the original structure for simulating a link failure event  $i$ . The resulting network is referred here as an incomplete network  $i$ . Note that there exists a total of  $L$  different incomplete networks

Step 2: Apply algorithms of VWP or WP to the incomplete network  $i$  and determine the number of fibers required for each fiber link, a total of  $L-1$  results. These numbers are then memorized in row  $i$  of an  $L \times L$  matrix  $\mathbf{R}$ . Each row  $i$  in the matrix  $\mathbf{R}$  contains results from each failure event  $i$ . Each column  $j$  contains the number of fibers required on link  $j$  for each failure event

Step 3: Repeat step 1 until all link failure events have been considered

Step 4: The maximum value of column  $j$  in the matrix  $\mathbf{R}$  is taken as the number of fibers required to place on link  $j$ . Therefore, the total number of fibers required for full protection is the summation of the largest value in each column

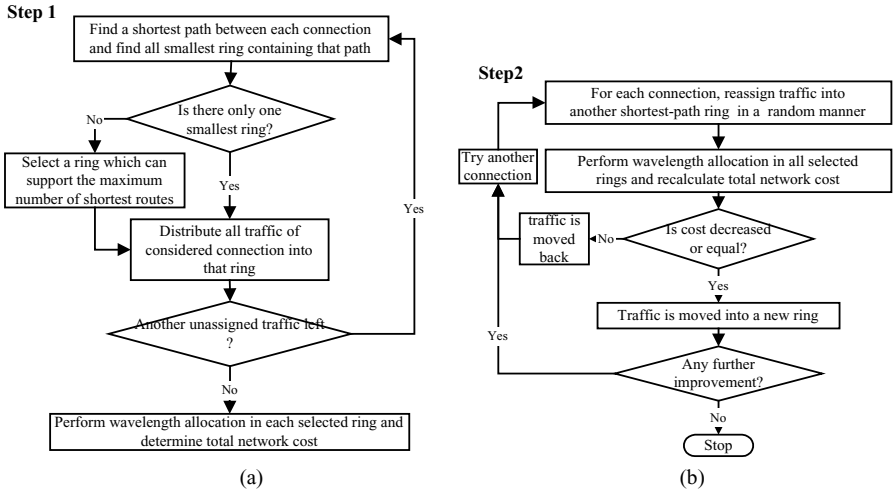
Step 5: The matrix  $\mathbf{R}$  obtained from the above steps is used as an initial result for further improvement. Since the network cost depends on the largest value in each column, it is useful to reduce the largest value of a certain column while keeping the largest values in all other columns unchanged. This technique can be repeated iteratively, until no further cost reduction is observed

### 3 Multi-ring Protection Technique

The design approach uses a heuristic algorithm which can be divided into two main steps. In the first step, ring selection and wavelength allocation are performed. The second step improves the design in order to minimize cost of the network and increase wavelength utilization in selected ring while getting rid of those rings that have low wavelength utilization. This step is iteratively performed until a satisfied solution is obtained, or no further improvement is observed. The details of each step are given as follows:



Step 1: Figure 3 (a) depicts the flowchart of step 1 process. Ring selection is a process in which an appropriate small set of rings is selected from all possible rings for handling all traffic demands. For each connection, a criterion used in the algorithm is that only one ring is selected to support all traffic of connection. The ring must include the shortest route between that node pair and should also be of the smallest size possible since a smaller ring is found to be more efficient in terms of bandwidth usage. Note that a ring provides two disjoint paths between each connection. Selecting the shortest ring that contains the shortest route ensures that traffic demands are transported over the shortest distance. This is similar to the mesh design, in which high resource utilization could be achieved. If there is more than one appropriate ring, a ring which accommodates the highest number of shortest routes of all connections is selected in order to minimize the number of selected rings. This process is performed on each connection until there is no further unassigned connection. Next, the wavelength allocation is accomplished ring-by-ring by algorithms H3 and H4 proposed in [8]. The number of fibers required on each ring can then be obtained from  $\lceil N_\lambda / M \rceil$  where  $N_\lambda$  is the number of maximum wavelengths required in all links and  $M$  is as defined previously. The network cost can be obtained from the summation of each selected ring cost which is the number of required fibers times the number of nodes in that ring



**Fig. 3.** (a) Procedure of the ring selection and wavelength allocation (b) Procedure of the design improvement

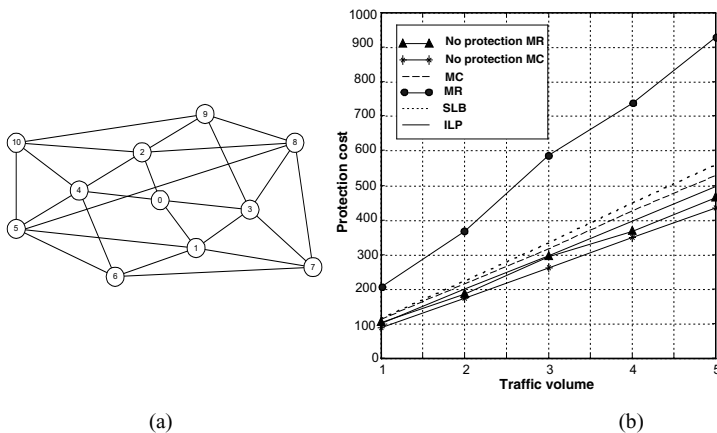
Step 2: In general, the first solution may not always be satisfactory because the number of selected rings could be very large. In an extreme case, the total number of rings in the network can be equal to the number of connections because the traffic of each connection is exclusively supported by an individual ring. Under this condition, the total network cost will be very expensive. Therefore, it is used as an initial result which needs further improvement using the following process. The traffic demand between each connection is considered again in turn to see whether other alternative

rings would result in a more effective allocation. In each round of improvements, the traffic between only one connection is considered. We attempt to move this traffic from pre-selected ring into a random alternative ring. For each connection, a candidate ring is selected randomly and must be a member of previously selected ring set. If an alternative ring produces similar or better results, the new design outcome is accepted as the current solution before performing the next iteration. Otherwise, the traffic will be moved back to the old ring. This procedure is repeated until all traffic demands are considered and no further improvement is observed. The process described above is shown in Figure 3 (b).

## 4 Simulation Results and Discussion

### 4.1 Protection Cost

A network topology from [9] is used to illustrate the protection costs of mesh and multi-ring approaches with varying values of  $M$  (1,2,4,8), see Figure 4 (a). This network which is referred to here as Euro-Core is deliberately selected for discussion purposes in this paper, because it has relatively small number of nodes ( $N = 11$  nodes) such that the ILP can still obtain the bound on the protection cost within a reasonable period of time. In addition, the network has very high level of connectivity, allowing a clear distinction between the costs of both design approaches. The traffic demands are assumed uniform with five levels of traffic volumes, *i.e.* 1 to 5. Therefore, the traffic volume of level  $i$  is defined as  $i$  times the total number of active paths among all node pairs, *i.e.*  $i \times N(N-1)/2$  active paths.



**Fig. 4.** (a) Experimental network: Euro-Core ( $N=11$ ,  $L=25$ ) (b) the protection cost with  $M=1$

Figure 4 (b) shows the protection cost of every approach with  $M=1$ . Note that when  $M=1$  there is no difference between the WP and VWP systems. Let first look at

the bound on the total network cost obtained from the ILP formulation and the cost of the MC design scheme achieved by the heuristic algorithm. It appears that the network costs with full protection from the MC design are slightly higher than the lower bound for all traffic volumes. This implies that the heuristic algorithm is quite effective for this network sample. When comparing the network costs of the MC and SLB designs (the SLB network cost is obtained from the heuristic algorithm in [7]), it is found that the differences between them are marginal, meaning that the SLB scheme can be made as cost-effective as the MC.

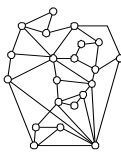
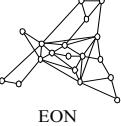
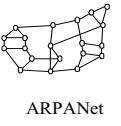
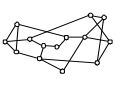
For the multi-ring protection technique, it is interesting to see that when no protection is provided the network cost (No protection MR) is just a little higher than that of the MC mesh approach (No protection MC). Note that the network cost of the MC approach is easily obtained from the summation of the number of lowest hop count of all node pairs; this is truly a minimal cost. The heuristic algorithm used for the MR design is therefore considered effective, because it is able to optimize the path allocation, such that both costs are comparable. The additional cost of the MR in respect to that of the MC is caused by the constraint that connections between a node pair must be confined only on one selected ring. For a given set of selected rings a pattern of traffic distribution, the multi-ring algorithm will attempt to balance the traffic load on every link of each individual ring, so that the number of fibers in the rings are minimized. To achieve this, some traffic loads will have to be assigned on a longer route.

We now turn to consider the differences between the MR and MC design approaches when protection is provided. For the multi-ring, it is a well-known fact that 100 % extra capacity is always needed for full protection. For the MC mesh the amount of extra capacity depends on the network topology and its connectivity in relation to traffic patterns. Usually the higher the connectivity is the lower the extra capacity will result. As mentioned earlier that this network sample has rather high connectivity, *i.e.* each node has on average 4.54 links adjacent to it, it is expected that the amount of extra capacity should be rather small. The heuristic algorithm used for finding the protection cost for the MC scheme described in this paper was able to achieve very good results, as the protection cost is only approximately 22.93% over the unprotected network cost. This particular example is certainly in favor of the MC mesh design, as it is much more expensive (up to 78.09%) to employ the multi-ring protection scheme in comparison to the MC mesh design. However, in a practical system which does not require 100% protection, the level of protection can easily be scaled to the desired level by removing some protection rings without changing or rerouting the existing design; this is an advantage of the multi-ring design over the mesh design.

In order to demonstrate further the differences between the protection costs of the multi-ring design and the MC mesh design, the results of several different network scenarios are given in Table 1. The traffic demands assumed in these networks are uniform with a volume of one and only one wavelength is multiplexed in each fiber. In the NSFNet, the multi-ring design requires 55.31% more capacity over the MC mesh counterpart, whereas in the ARPANet the difference is even smaller, *i.e.* only 40.74%. These two network configurations show a closer gap between the costs of the multi-ring and the mesh design when full protection against single link failures is provided. It is interesting to point out that both network structures have almost the same average number of links adjacent to a node *i.e.* 3, meaning that they both should

theoretically require spare capacity of about 50%. This figure is obtained from a simple calculation. If a failure occurs at a certain link of a node, the traffic carried over that link would have to be redirected through the remaining 2 links connected to the node. If these disrupted traffic loads are split equally over the two links, each of the two links will have to provide an extra capacity of 50% of the normal traffic. It turns out that the ARPANet demands more than 50% spare capacity (54.14%) whereas the reverse is true for the NSFNet (40.00%). As the costs of the multi-ring and MC mesh are not very different when no protection is included, it means that the total cost differences between the multi-ring and the MC mesh will depend on that how much each individual topology can take advantage on the alternative routes under a network failure.

**Table 1.** The network costs with and without protection achieved from the MC and MR heuristic algorithms. The value of  $C$  is the average number of links adjacent to a node. The percentage of difference between the cost with and without protection of MC is given in parenthesis

Network	$N$	$L$	$C$	MC		MR		$\frac{(MR-MC) \cdot 100}{MC}$ (with protection)
				without protection	with protection	without protection	with protection	
 UKNet	21	39	3.71	526	732 (39.16%)	597	1194	63.11%
 EON	18	35	3.89	336	505 (50.29%)	394	788	56.04%
 ARPANet	20	31	3.10	543	837 (54.14%)	589	1178	40.74%
 NSFNet	14	21	3.00	195	273 (40.00%)	212	424	55.31%

Let now turn back to our EURO-core example. Figure 5 illustrates the protection cost of the VWP system as a function of  $M$  and in this case the traffic volume is set to four. The network cost increases as the value of  $M$  increases. This means that, for each link of the network, the fibers have many channels that are unassigned wavelength for communications (in order words, the efficiency of fiber utilization of that link is low). As the value of  $M$  gets large, the number of unassigned channels becomes greater. Moreover, in multi-ring scheme, all links in a selected ring occupied

the same number of fibers, which is the maximum fiber requirement of all links. Thus, the utilization of fibers in multi-ring scheme is certainly less efficient than the mesh design.

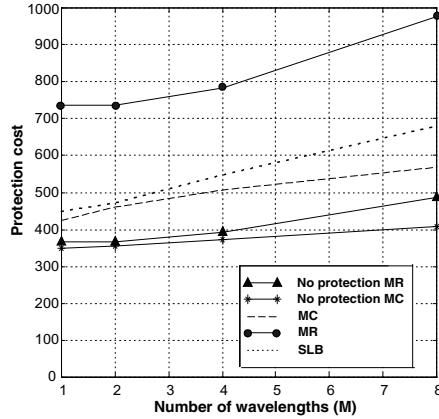


Fig. 5. The VWP protection cost with various values of M

Let us focus on the effect of M and the traffic demands on the ring utilization. To illustrate this, the number of selected rings is chosen to indicate the ring utilization as shown in Figure 6 (a). When the traffic demand is set at low level with a value of M greater than 1 (such as at point A), if the traffic loads are distributed among many separated rings, each ring will have a very low link utilization. Therefore, it is useful and more effective to aggregate these traffic loads into a smaller number of rings in order to increase the utilization of these chosen rings. When traffic is higher, each fiber of ring is more filled up and has a high utilization even if the traffic loads are distributed on many separated rings. Therefore, the higher traffic demand at this point results in a higher number of selected rings (such as at point B). This is true until the traffic is high enough that there are some remaining traffic loads which may need to be assigned on a new fiber. The ring utilization will be at a low level as in the case of a low traffic if the traffic loads are distributed among many rings. To maximize the ring utilization, a small number of rings are selected and this results in a decreased value of selected rings again (such as at point C). As seen from Figure 6 (a), the number of rings is repeated in this manner periodically. Moreover, the longer period at a higher value of M implies that the value of ring utilization changes more slowly when we multiplex more wavelengths into the fiber.

Here we investigate the effects of wavelength conversion. In the mesh designs, the ratio of the total network costs between the WP and VWP systems tend to increase with the number of wavelengths multiplexed in each fiber (M); this is depicted in Figure 6 (b). This highlights the importance of wavelength conversion when using a higher number of wavelengths in a fiber. This is particularly obvious in the SLB mesh technique. A reduction of 20-35% in the network cost is observed with the values of M between 4 to 8. On the other hand, in the multi-ring technique, wavelength conversion has so little influence. Since conversion is not useful for wavelength

allocation in a single ring network as described in [8], no cost savings will be accomplished.

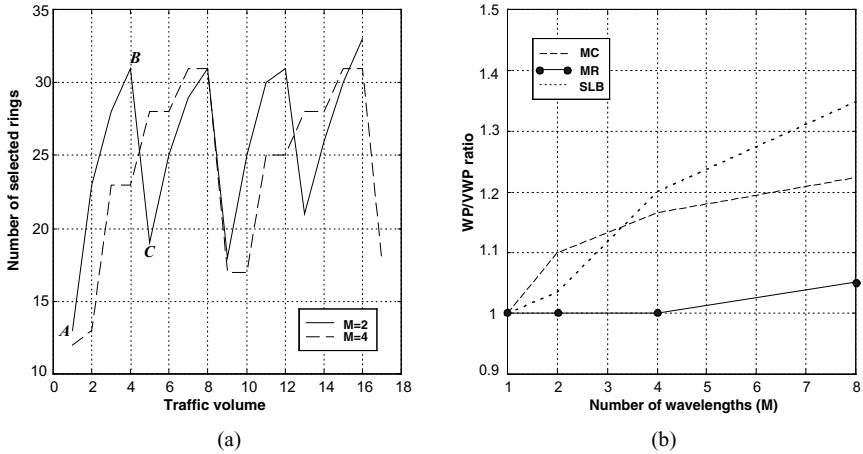


Fig. 6. (a) Relationship between number of selected rings and traffic volume (b) Difference between WP and VWP systems

## 4.2 Execution Time Requirement of the ILP Technique

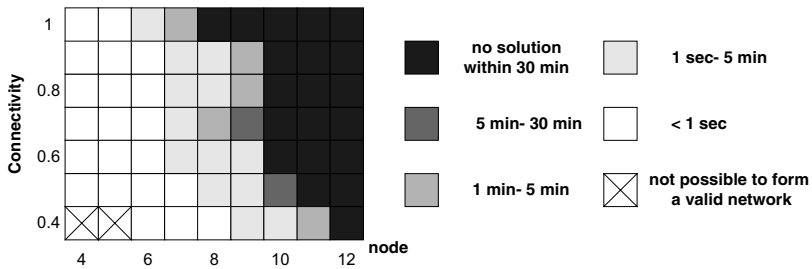


Fig. 7. The computational time for the ILP technique

A chart of the computational time requirement of the ILP technique of the MC approach is summarized in Figure 7. The execution results here are selected from test samples and the traffic patterns are uniform. Experimental network topologies are presented in term of the number of nodes and the physical network connectivity. The network connectivity is defined as the ratio of the number of links of testing networks to a full-connected network of the same number of nodes [9]. Based on these results, the execution time is longer as the connectivity or the number of nodes increases (corresponding to the large problem formulation). Therefore, the ILP technique is not practical when the network is greater. For example, the solution cannot be found within 30 minutes when the number of nodes is more than 10.

## 5 Conclusions

Strategies for designing WDM transport network against the single link failure based on the multi-ring and mesh designs have been discussed. In the mesh network, from the concept of MC scheme, integer linear programming (ILP) and heuristic algorithms are used to evaluate the protection cost. The complexity of ILP, however, depends on the network size, and the result is used as a lower bound of protection cost. The comparison between MC and SLB costs shows that the improvement due to the former is marginal. It is clear that the SLB scheme is sufficient for finding the minimal cost. In the multi-ring design, a heuristic algorithm is proposed. It was found that the protection cost of the multi-ring design is higher than that of the mesh design. However, in the case when full protection cost of multi-ring is not required, there are substantial benefits.

Based on our simulation, it is found that the number of wavelengths multiplexing into a fiber ( $M$ ) plays a major role on the network protection cost in both mesh and ring design. Wavelength conversion affects the protection cost in the mesh design especially with a high value of  $M$ . In multi-ring design, It does not significantly reduce the protection cost.

## References

1. R. Ramaswami and K. Sivarajan, "Optimal Routing and Wavelength Assignment in All-Optical Networks," in *Proc. INFOCOM'94*, June 1994.
2. D. Banerjee and B. Mukherjee, "A Practical Approach for Routing and Wavelength Assignment in Large Wavelength-Routed Optical Networks," *IEEE Journal on Selected Areas in Communications*, vol. 14, pp. 903-908, June 1996.
3. N. Wauters and P. Demeester, "Design of the Optical Path Layer in Multi-wavelength Cross-connected Networks," *IEEE Journal on Selected Areas in Communications*, vol. 14, June 1996.
4. S. Ramamurthy and B. mukherjee, "Survivable WDM Mesh Networks, Part I," in *Proc. INFOCOM'99*, vol. 2, pp. 744-754, 1999.
5. L. Wuttisittikulkij and M. J. O'Mahony, "Use of Spare Wavelength for Traffic Restoration in Multi-Wavelength Transport Network," In *IEEE Proceedings of ICC'96*, pp. 1778-1782, Texas, 1996.
6. L. Wuttisittikulkij and M. J. O'Mahony, "Design of a WDM Network using a Multi-ring Approach," in *IEEE Proceedings of GLOBECOM'97*, pp. 551-555, November 1997.
7. N. Nagatsu and K. Sato, "Optical Path Accommodation Considering Failure Restoration with Minimum Cross-connect System Scale," *IEEE/Network Operation and Network Symposium '96*, 1996.
8. L. Wuttisittikulkij, S. Leelanunnukul, S. Arreewanit and P. Prapinmongkolkarn, "Routing and Wavelength Allocation in Multi-wavelength All-optical Ring Networks," *IEEE Proceedings of ICC'99*, vol. 3, pp. 2018-2022, June 1999.
9. S. Boroni, P. Bayvel and Richard J. Gibbens, "On the Number of Wavelengths in Arbitrarily-Connected Wavelength-Routed Optical Networks," in *OSA Trends in Optics and Photonics Series (TOPS): Optical Network and Their Applications*, vol. 20, pp. 195-204, July 1998.

# Membership-Insensitive Totally Ordered Multicast: Properties and Performance

Jerzy Konorski<sup>1</sup>

<sup>1</sup>Technical University of Gdansk  
ul. Narutowicza 11/12, 80-952 Gdansk, Poland  
jekon@pg.gda.pl

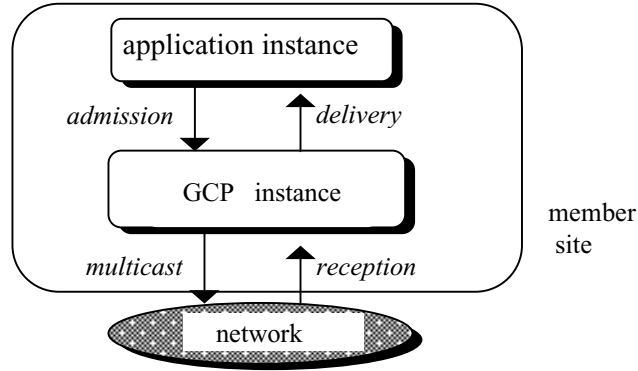
**Abstract.** Membership-insensitive group transport protocols are particularly desirable for large groups of fast-varying membership where the distributed application being served insists on collective output and survivability rather than producing a consistent record of successive group views. An overview of such a protocol, able to maintain agreed delivery order of multicast messages across the group except for scenarios that can be made arbitrarily improbable, is presented along with a proposed specification of the so-called '1C network service with ContiguityDetector' it is built upon. Key properties of the protocol are expressed through some graph-theoretic observations. LAN implementation experience and performance measurements are described to conclude that the group throughput remains high under constant group membership and, there being virtually no group reconfiguration overhead, varies gracefully in step with the number of active group members.

## 1 Group Communication Model

Group communication models extend the familiar point-to-point (unicast) paradigm by introducing

- a family of patterns of message transfer, involving a group of member sites rather than just two, of which multipoint-to-multipoint is the closest to the intuitive notion of group communication (the related *multicast protocols* [5] are focused on typically best-effort multicast routing and switching within the underlying communication network),
- a wider choice of QoS options that include, besides performance-oriented ones, group-wide consistency constraints like reliability, message ordering and membership control (the related *group transport protocols* [13] focus on implementation of these constraints, typically through multiple point-to-multipoint message transfers with coupled control); when referring to a *group communication protocol* (GCP) we shall understand a group transport protocol, the multicast protocol functionality being left to the underlying network service.





**Fig. 1.** Logical placement of a GCP instance

Logically, as Fig. 1 shows, a GCP instance at a member site is situated between the local instance of the distributed application it serves and the network service it uses.<sup>1</sup> We will carefully distinguish between message admission and delivery, and message multicast and reception, actions occurring across the application-to-GCP and GCP-to-network interfaces, respectively. With regard to the elements of the model in Fig. 1 we state the following assumptions and non-assumptions:

- **application instances:** generate messages at will, as determined by the (arbitrary) course of application execution; each message is destined for the whole of the group consisting of  $N$  member sites (we consider one group only),
- **GCP instance:** multicasts messages it has admitted, receives messages as they arrive from the network and delivers them to the application instance subject to prescribed group-wide consistency constraints.
- **network service:** employs any best-effort multicast protocol thus constituting an asynchronous environment with arbitrary message delays and losses.
- **sites:** may exhibit intermittent presence i.e., undergo alternate presence and absence spells (e.g., due to outages or switching to other tasks), thus constituting a potentially fast-varying membership with non-Byzantine site faults permitted.

The rest of the paper is organised as follows. In Sec. 2 the relevant consistency constraints are specified and a case is made for membership-insensitive GCPs. A suitable network service model, called 1C with ContiguityDetector, is introduced in Sec. 3 along with a few exemplary network settings. A membership-insensitive GCP is proposed in Sec. 4 and some key properties thereof stated in Sec. 5. Finally, Sec. 6 describes a LAN implementation experience with the proposed protocol and concludes with a brief discussion of its measured performance.

<sup>1</sup> Fig. 1 need not map directly onto the ISO-OSI layering e.g., ‘GCP instance’ may be either part of the transport or application layer or split between the two; accordingly, ‘network’ may or may not encompass transport layer functionality.

## 2 Consistency Constraints

The group-wide consistency constraints a GCP has to meet, Agreed Delivery Order (ADO) and Membership Control, are very different in nature, which gives rise to splitting a GCP into separate subprotocols. Although group communication need not be connection-oriented, we may use the term *group connection* to designate a self-contained exchange of a set  $M$  of multicast messages. Let the member sites be numbered 1 through  $N$  and let  $R_n$  and  $D_n$  be the sequences of messages respectively received and delivered at site  $n$  throughout the group connection. ADO states that, regardless of the  $R_n$ 's, the following holds for the  $D_n$ 's:

- **Atomicity:** for any  $m \in M$ ,  $m \in D_{n^*}$  at some site  $n^*$  implies that  $m \in D_n$  for all  $n$ ,
- **Total Order:** a total order ' $\rightarrow$ ' on  $M$  can be produced such that for any site  $n$  and  $m, m' \in M$ , ( $m, m' \in D_n$  and  $m \rightarrow m'$ ) implies that  $m$  precedes  $m'$  in  $D_n$ .

Note that neither of these statements implies the other; in conjunction they permit a distributed application to run as if supplied from a central event queue.<sup>2</sup>

Preserving ADO clearly requires that the member sites perceive themselves as members. Given that some of them may enter an absence spell at times, one wants to confine the above ADO specification to the constant membership case. The opposite case is covered by Membership Control. Let  $e_{n,i}$  be the  $i^{\text{th}}$  membership change event perceived at site  $n$  and let  $V(e_{n,i})$  be the membership view implied by  $e_{n,i}$  i.e., the set of sites perceived by site  $n$  as present upon  $e_{n,i}$ . Then the following is to hold:

- **Membership Consensus:** all sites  $n' \in V(e_{n,i})$  perceive  $e_{n,i}$ ,
- **Virtual Synchrony** [2]: all sites  $n \in V(e_{n,i}) \cap V(e_{n,i+1})$  i.e., members of two consecutive membership views, deliver messages in ADO between perceiving  $e_{n,i}$  and  $e_{n,i+1}$ ; **Extended Virtual Synchrony** [11] enables to resume Virtual Synchrony when some previously absent sites have entered another presence spell.

ADO is costly in terms of message latencies and overhead, which may behave designers to consider quasi-ADO protocols instead, where violations of ADO are possible though materialise arbitrarily rarely. Membership Control, besides the well-known impossibility result [8], fails in that ultimately it is to unify the perception of successive group views, which is not a universal motivation. It is for collective accountancy-oriented applications (banking, reservation systems), but not for collective output-oriented ones, especially involving large and/or variable groups, where membership sensitivity is more a burden than a benefit. We illustrate this point by an example of a generic distributed resource sharing framework.<sup>3</sup>

<sup>2</sup> Some authors [16] add another dimension by distinguishing weak and strong Total Order. Others [9] define Validity and Integrity to ensure eventual delivery of multicast messages and prevent delivery of fake ones, respectively. The former is covered by the Eventual Receipt assumption (Sec. 4); the latter is only important for Byzantine site faults.

<sup>3</sup> A similar point can be made for some other parallel tasking frameworks e.g., parallel simulation with rollback.

Suppose that the member sites are to perform successive operations on a countable reusable resource and that the throughput (total operations per second) determines the output we get. Temporary exclusive control over resource units is transferred among the sites via multicast request, allocation and release messages, the former processed at each site in the FIFO order. To prevent the so-called wait-for condition and the resulting possibility of deadlock, a site that has not been allocated enough resource units for its operation is obliged, upon expiry of a time-out, to release the collected resource units and start requesting anew. Preserving ADO of request and allocation messages is important as any inconsistency would favour a situation of two or more sites collecting resource units in parallel and ending up with expired time-outs, which would reduce the throughput we are interested in maximising. Two observations emerge from the above example:

- Violation of ADO, although worsens the performance, need not be catastrophic to the application. Ad-hoc simulation experiments performed for several realistic WAN settings typically revealed a mere 10% operation throughput drop with as much as 5% messages violating ADO, which suggests there is a need for quasi-ADO protocols given the high cost of maintaining strict ADO in large groups.
- The sites are completely uninterested in tracking current membership. In fact, some of them may get uninterested at all and enter an absence spell; subsequently they may return and when they do, they are to promptly resume (quasi-)ADO instead of triggering costly rejoin-group procedures. For the TOTEM GCP [1], an ad-hoc simulation study shows that a single site out of 20, toggling periodically between presence and absence, causes reconfiguration overhead that halts the group connection for about 20% of the time and reduces the group throughput roughly by half for another 20%. Thus there is a need for membership-insensitive GCPs even if they are quasi-ADO rather than ADO. Note that membership insensitivity increases survivability in a non-protective or hostile environment.

### 3 Network Service Model

GCP design requires specification of the underlying multicast service. Some well-known ADO protocols are built on top of an unreliable service [1], [15], reliable service [14] or reliable FIFO service [7]; quite a few rely on causal service [2], [6].<sup>4</sup> Single-access broadcast media (e.g., Ethernet, FDDI, Token Ring, single-channel wireless TDMA) make a favourable environment since their hardware- and MAC-level broadcast and message ordering naturally define ' $\rightarrow$ ' as the transmission order. Some ADO protocols make explicit use of this property [10], [12]. An appropriate service abstraction goes by the name of *1-channel (1C) service* [12] and can be reformulated analogously to Total Order using the  $R_n$ 's rather than  $D_n$ 's:

---

<sup>4</sup> This does not contradict the standard argument that message ordering should be an application layer issue [4] since both reliable FIFO and causal multicast services can be emulated within a GCP as a form of ADO preprocessing.

- **1C:** a total order  $\rightarrow$  on  $M$  can be produced such that for any site  $n$  and messages  $m, m' \in M$ ,  $(m, m' \in R_n \text{ and } m \rightarrow m')$  implies that  $m$  precedes  $m'$  in  $R_n$ .

1C is not a reliable multicast service: due to possible reception misses (caused by transmission losses, buffer overflow etc.), Atomicity cannot be guaranteed at the network level. Nevertheless, a realistic performance-oriented enhancement of the 1C service is possible. Let  $\Rightarrow$  denote contiguity in  $\rightarrow$  i.e.,  $m \Rightarrow m'$  means that  $m \rightarrow m'$  and there is no  $m''$  such that  $m \rightarrow m''$  and  $m'' \rightarrow m'$ . At each site an inference device called ContiguityDetector is placed that for any pair of consecutively received messages  $m$  and  $m'$  infers either  $m \rightarrow m'$  or  $m \Rightarrow m'$  as specified below:

- if  $m \Rightarrow m'$  is inferred at a site then indeed  $m \Rightarrow m'$ ,
- if  $m \Rightarrow m'$  then the inference about  $m$  and  $m'$  ( $m \rightarrow m'$  or  $m \Rightarrow m'$ ) may vary from site to site depending on local conditions.

We now discuss several realistic settings that virtually provide the above service.

### 3.1 Single-Access LAN

Messages arriving at a site are stored in a reception buffer whence they are fetched by the local GCP instance for processing. Slow-fetch and the resulting message overwrite can be assumed the sole factor behind reception misses; other factors (e.g., transmission errors, receiver overruns, software bugs) are marginalised over time by the advances in LAN technology. The ContiguityDetector at a site reduces to a bit set whenever an arriving message overwrites a previous one in the reception buffer, and reset after reception of any message; if the bit is found reset at the instant of message  $m'$  reception then  $m \Rightarrow m'$  is inferred,  $m$  being the latest message received. Alternatively, a timing-based ContiguityDetector infers  $m \Rightarrow m'$  if the receptions of messages  $m$  and  $m'$  are less apart in time than what is needed to transmit a message.

### 3.2 Bounded-Delay WAN

Upper-bounding message delays by a  $\Delta$  converts the network into a synchronous environment suitable for timed ADO multicast service [9]. Assuming that local clocks at sites run synchronously, messages are delivered in ADO based on increasing reception timestamps. By adding  $\Delta$  to the current time, a sender obtains the reception timestamp for its message, whose lifetime is limited to  $\Delta$ . To avoid excessive delivery latencies, a  $\Delta'$  can be used instead ( $\Delta' < \Delta$ ) at the cost of some messages using up their lifetime before reaching all the member sites. The ContiguityDetector at site  $n$  analyses current network delays to site  $n$  and for a received message  $m'$  determines whether the delays have remained under  $\Delta'$  since the latest message  $m$  was received. In that case,  $m \Rightarrow m'$  is inferred.

### 3.3 Tag Sequencer

Messages are tagged with unique identifiers and use a reliable multicast service. A copy of the tag is unicast to a special sequencer site which numbers arriving tags sequentially and multicasts them back to the group. At any site, messages whose numbered tags have been received are delivered in sequence. For security and/or privacy reasons, tags from different groups should appear indistinguishable to the sequencer; it must therefore use a broadcast rather than multicast service for numbered tags e.g., best-effort FIFO broadcast. Now multiple groups share the same tag sequence, thus when a site has filtered out other groups' tags then for two consecutive tags numbered  $i$  and  $j$  ( $i < j$ ) corresponding to messages  $m$  and  $m'$ , it can infer  $m \Rightarrow m'$  only if the filtered out tags' numbers are contiguous between  $i+1$  to  $j-1$ .

## 4 Distributed Precedence Graph Protocol

We now present a survivable membership-insensitive GCP named Distributed Precedence Graph (DPG) that operates on top of the 1C service with ContiguityDetector. The network service is also assumed to fulfil Eventual Receipt [3] i.e., infinitely many message remulticasts result in infinitely many receptions at each member site. The DPG protocol exhibits quasi-ADO and Extended Virtual Synchrony in that, except for scenarios that occur arbitrarily rarely, each member site delivers continuous segments of the ADO sequence in successive presence spells.

A generic DPG instance at site  $n$  will be denoted DPG- $n$  for short. The basic data structures at DPG- $n$  consist of admission and ordering buffers at the application-to-DPG interface, across which messages are exchanged, as well as a multicast queue, a reception buffer and the ContiguityDetector at the DPG-to-network interface, across which PDUs are exchanged. Two PDU types are distinguished:

- '*message*' PDU - contains a new message (multicast for the first time) along with a unique message tag; the tag is valid only for the message's lifetime and need not reflect any static site numbering nor message sequencing within a site,
- '*report*' PDU - helps construct ADO, recover lost messages, perform flow control and local clock rate synchronisation.

A local clock controls the multicast interval (for flow control), remulticast time-outs (for recovery of lost messages), as well as deadlines for message incorporation into ADO (to implicitly eliminate absent sites).

### 4.1 Construction of Quasi-ADO

DPG- $n$  maintains an acyclic directed *precedence graph*  $\mathbf{G}$  whose vertices map onto message tags (written symbolically  $X$ ,  $Y$ ,  $Z$  etc.) and arcs reflect the order ' $\rightarrow$ '.  $X \in \mathbf{G}$  implies that DPG- $n$  has received message  $X$  or has been notified of its reception at some other site via a '*report*' PDU.  $(X, Y) \in \mathbf{G}$  means that messages  $X$  and  $Y$  have been

consecutively received at some site, implying  $X \rightarrow Y$  though not necessarily  $X \Rightarrow Y$ . Reception of message  $Z$  causes the inclusion into  $\mathbf{G}$  of vertex  $Z$  and arcs  $(X, Z)$  for all  $X$  having no successor in  $\mathbf{G}$  (while the message body is stored in an ordering buffer). Moreover, DPG- $n$  maintains the tag ( $L\_REC$ ) of the latest received message and marks arc  $(L\_REC, Z)$  if  $L\_REC \Rightarrow Z$  is inferred at the instant of message  $Z$  reception. DPG- $n$  includes in 'report' PDUs it issues a subgraph of  $\mathbf{G}$  corresponding to messages not yet delivered locally. Supposing DPG- $n$  has received graph  $\mathbf{G}'$  in a 'report' PDU, first it computes  $\mathbf{G} := \mathbf{G} \cup \mathbf{G}'$  and marks each arc that was marked in  $\mathbf{G}$  or  $\mathbf{G}'$ . Next,  $\mathbf{G}$  is *linearised* so as to take advantage of the newly acquired knowledge of ' $\rightarrow$ '. Linearisation continues as long as, for any arc  $(X, Y)$ ,

- there exists a vertex  $Z \in \mathbf{G}$  such that either  $(X, Z) \in \mathbf{G}$  and  $(X, Z)$  is marked or  $(Z, Y) \in \mathbf{G}$  and  $(Z, Y)$  is marked;  $(X, Y)$  is then replaced by  $(Z, Y)$  or  $(X, Z)$  respectively;
- there exists a directed path from  $X$  to  $Y$  in  $\mathbf{G}$  consisting of more than one arc;  $(X, Y)$  is then removed from  $\mathbf{G}$ .

## 4.2 Message Incorporation into Quasi-ADO

Two vertices of  $\mathbf{G}$ ,  $E\_DELIV$  and  $L\_DELIV$ , correspond to the earliest and latest deliverable messages (already incorporated into quasi-ADO). Message  $X$  becomes deliverable when the arc  $(L\_DELIV, X)$  is marked; when all immediate successors of  $L\_DELIV$  have been included in  $\mathbf{G}$  for at least *Deliv\_Cycle* seconds, they become deliverable in any static order e.g., of decreasing tags. Local delivery of  $E\_DELIV$  follows when either the corresponding message body is found in the ordering buffers or *Deliv\_Cycle* seconds have elapsed since it became deliverable (in the latter case a 'dummy' message is delivered with the understanding that all protocol instances that have the original message are currently absent).

## 4.3 Lost Message Recovery

DPG- $n$  appends a list  $LM$  of lost message tags to issued 'report' PDUs as well as a list  $RM$  of messages (tags and bodies) remulticast in response to earlier received 'report' PDUs. Message  $X$  loss is detected upon: (1) reception of a 'report' PDU with graph  $\mathbf{G}'$  containing a vertex  $X \notin \mathbf{G}$  and/or a remulticast message  $X$  not found in the ordering buffers, which cannot be stored because of overflow, or (2) reception of a 'message' PDU while the ordering buffers overflow. In case (2), all the ordering buffers might be occupied by successors of message  $X$ . To prevent such a deadlock, message  $X$  is permitted to seize the ordering buffer occupied by any successor  $Y$ , and the latter is lost instead of message  $X$ . Accordingly,  $X$  or  $Y$  is put on  $LM$  in the next issued 'report' PDU to act as a remulticast request. If necessary, remulticast requests are repeated every *Retr\_Cycle* seconds.

#### 4.4 'Report' PDU Heartbeat

The protocol attempts to maintain a uniform 'report' PDU cycle throughout the group. To do that, upon reception of some other protocol instance's 'report' PDU, DPG- $n$  delays the multicast of its own for *Report\_Cycle* seconds. However, DPG- $n$  can waive this delay if graph  $\mathbf{G}'$  in the received 'report' PDU (say issued by DPG- $n$ ) contains an arc that would be replaced or removed in  $\mathbf{G} \cup \mathbf{G}'$ . In that case DPG- $n$  views the current knowledge of ' $\rightarrow$ ' at DPG- $n$ ' as incomplete regardless of the fact that some messages recently received by DPG- $n$  may not have been received and accounted for at DPG- $n$ ' at the instant it issued the 'report' PDU.

#### 4.5 Flow Control

The flow control mechanism in DPG- $n$  is supposed to counteract ordering buffers overflow and premature removal of messages from  $\mathbf{G}$ . Locally delivered messages concluded to have also been delivered at all currently present sites are removed from  $\mathbf{G}$  at a limited rate depending on the current throughput bottleneck estimate, called *Message\_Cycle*, that determines the minimum interval between successive multicasts of 'message' PDUs. *Message\_Cycle* is contained in 'report' PDUs and incremented by a fixed amount *Delta* whenever an ordering buffer at DPG- $n$  overflows; it is set to  $\max\{\text{Message\_Cycle}, \text{Message\_Cycle}\}$  upon reception of a 'report' PDU containing *Message\_Cycle'*, and decremented by *Delta* periodically every *Fc\_Cycle* seconds (down to a predetermined lower bound).

### 5 Protocol Properties

Typically, a message  $X$  becomes deliverable upon marking the arc  $(L\_DELIV, X)$ , which means  $L\_DELIV \Rightarrow X$  was inferred at some member site. The other possibility i.e., all immediate successors of  $L\_DELIV$  reaching the limit of *Deliv\_Cycle*, implies that *Deliv\_Cycle* should be large enough to allow formation of identical successor sets at each member site. Granted that, the question is whether there is enough inter-site consistency regarding successors of  $L\_DELIV$ . Observations 1 through 3 below state that  $\mathbf{G}$  remains consistent with ' $\rightarrow$ ' across a succession of linearisations. Observations 4 and 5 are conjectures based on the protocol specification and simulation, yet to be proved rigorously.  $\mathbf{G}^+(X)$  and  $\mathbf{G}^-(X)$  denote the subgraphs of successors and predecessors of  $X$  in  $\mathbf{G}$ , and  $\text{lin}(\mathbf{G})$  the linearised graph.

**Observation 1.** Linearisation preserves consistency of  $\mathbf{G}^+(L\_DELIV)$  with ' $\rightarrow$ ' in that  $(X, Y) \in \mathbf{G}^+(L\_DELIV)$  implies  $X \rightarrow Y$ .

*Proof:* Suppose that  $(X_0, Y_0) \in \mathbf{G}^+(L\_DELIV)$  at DPG- $n$ . Observe that the marking of the arc  $(X_0, Y_0)$  would mean that at some site, messages  $X_0$  and  $Y_0$  were received

consecutively and without an intervening reception miss; consequently  $X_0 \Rightarrow Y_0$ . Assume then that arc  $(X_0, Y_0)$  is not marked, which leaves two possibilities:

1) Messages  $X_0$  and  $Y_0$  were received consecutively at some DPG- $n$  which thus included arc  $(X_0, Y_0)$  into its precedence graph  $\mathbf{G}'$  subsequently disseminated in a 'report' PDU. Then, by virtue of the 1C specification (Sec. 3),  $X_0 \rightarrow Y_0$ .

2) Arc  $(X_0, Y_0)$  was included into  $\mathbf{G}$  via replacing some arc  $(X_1, Y_0)$  or  $(X_0, Y_1)$ . Consequently,  $(X_1, X_0)$  is marked (i.e.,  $X_1 \Rightarrow X_0$ ) or  $(Y_0, Y_1)$  is marked (i.e.,  $Y_0 \Rightarrow Y_1$ ). Considering in turn how each of the two replaced arcs could have been included into  $\mathbf{G}$ , we have again two possibilities analogous to 1) and 2). Repeating this reasoning  $i+j$  times we can conclude the existence in  $\mathbf{G}$  of vertices  $X_0, X_1, \dots, X_i$  and  $Y_0, Y_1, \dots, Y_j$  such that  $X_i \Rightarrow \dots \Rightarrow X_1 \Rightarrow X_0$  and  $Y_0 \Rightarrow Y_1 \Rightarrow \dots \Rightarrow Y_j$  and  $(X_i, Y_j) \in \mathbf{G}^+(L\_DELIV)$ . Supposing that  $(X_i, Y_j)$  was included into  $\mathbf{G}$  via possibility 2), one should similarly deduce the existence of another vertex, either  $X_{i+1}$  or  $Y_{j+1}$ . Yet graph  $\mathbf{G}$  remains finite within a group connection, thus eventually for some  $i, j$  we will see that  $(X_i, Y_j)$  was included into  $\mathbf{G}$  via possibility 1). Therefore  $X_i \rightarrow Y_j$  and so  $X_0 \rightarrow Y_0$ .

**Observation 2.** Suppose that a reception of a 'message' PDU results in extending graph  $\mathbf{G}$  by a subgraph  $\mathbf{G}_e$ , then  $\text{lin}(\mathbf{G} \cup \mathbf{G}_e)$  contains  $\mathbf{G}$ . (This property guarantees that the local quasi-ADO stabilises in time despite reception of new messages.)

*Proof:* Suppose that DPG- $n$  has received message  $Z$ . Then arcs of the form  $(X, Z)$  are included into  $\mathbf{G}$ , where  $X \in \mathbf{G}$  and  $\mathbf{G}^+(X) = \emptyset$ . This may create a directed path to  $Z$  and so lead to removing some arcs, but only ones of the form  $(Y, Z)$  i.e., not in  $\mathbf{G}$ . If  $(L\_REC, Z)$  is marked then some arcs can be replaced, but in this case  $\mathbf{G}^+(L\_REC) = \emptyset$  and thus again only arcs of the form  $(X, Z)$  i.e., not in  $\mathbf{G}$ , are so endangered.

**Observation 3.** Linearisation is commutative in the sense that  $\text{lin}[\text{lin}(\mathbf{G} \cup \mathbf{G}') \cup \mathbf{G}''] = \text{lin}[\text{lin}(\mathbf{G} \cup \mathbf{G}'') \cup \mathbf{G}']$ . (This property guarantees that eventually, a locally obtained quasi-ADO is independent of the order of reception of 'report' PDUs.)

*Proof:* We show that the order of replacing or removing arcs is immaterial i.e., the arc set of  $\text{lin}(\mathbf{G})$  is fully determined by that of  $\mathbf{G}$ . From the proof of Observation 1 we conclude that  $(X_0, Y_0) \in \text{lin}(\mathbf{G})$  iff  $\mathbf{G}^+(X_0) \cap \mathbf{G}^-(Y_0) = \emptyset$  and there exist vertices  $X_1, \dots, X_i$  and  $Y_1, \dots, Y_j$  such that  $(X_i, Y_j) \in \mathbf{G}$ ,  $(X_k, X_{k-1})$  is marked, for all  $k=1..i$  and  $(Y_{k-1}, Y_k)$  is marked, for all  $k=1..j$ . Neither of these conditions has to do with the order of operations performed during a linearisation.

**Observation 4.** On return after an absence period, a protocol instance is free to resume the local construction of quasi-ADO once it has reset the connection status. (To minimise the risk of a message tag collision, prior reception of a number of 'report' PDUs is recommended.)

**Observation 5.** Given Eventual Receipt and proper DPG configuration, the  $D_n$ 's remain consistent with quasi-ADO for an arbitrarily long period of time with Extended Virtual Synchrony preserved.



## 6 Implementation and Performance Measurements

We shall focus on group throughput defined as the average message delivery rate throughout a group connection. Note that the DPG protocol (in fact, any GCP) largely reduces the PDU transport potential of the underlying network since construction of (quasi-)ADO across a large group is a more demanding task than providing a FIFO point-to-point service. To investigate the group throughput, it was decided to implement the DPG protocol in an available LAN environment.

DPG-*n* has been implemented as about 2000 lines of ANSI C code to run on a Pentium 100.333 MHz workstation under Linux (kernel version 2.0.36). Up to 17 such workstations were configured into a group interconnected by a switched 10 Mb/s Ethernet. The 1C service specification was found to hold, and the single-access LAN version of the ContiguityDetector was adopted. *Report\_Cycle* was set so as to keep the 'report' PDU traffic within 20% of the total message traffic, while the flow control was optimised towards maximum group throughput.

The implementation uses standard UDP broadcast interface, as shown in Fig. 2. One BSD socket is used for multicast and a separate one, configured as an asynchronous I/O port, for reception. At the top, a message generator/absorber inputs a stream of messages, thereby assuming the role of an application instance.

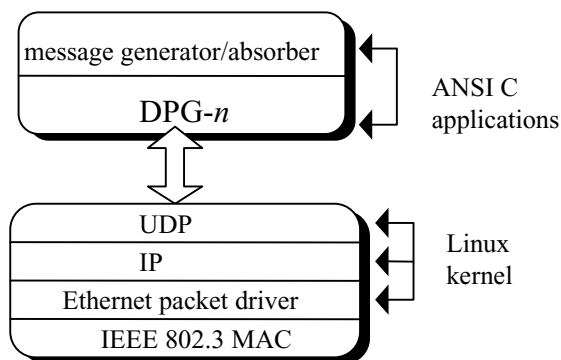


Fig. 2. DPG implementation layering

A few lessons (not necessarily applicable to WAN environments) have been learned regarding the DPG protocol performance in the specific LAN environment:

- Multiple reception buffers are needed instead of just one assumed for argument in Sec. 3.1 (20 in our implementation), with the ContiguityDetector inference rules appropriately redefined; the induced miss rate remains then well below 1%, and the resulting ADO violation rate is on order of one message in many thousands.
- Local clocks sometimes prove not accurate enough for flow control; in a recent version of our implementation, *Message\_Cycle*, *Deliv\_Cycle*, *Retr\_Cycle* and *Report\_Cycle* are all expressed in terms of the number of 'message' PDUs received since the event that triggered the respective timer.

- Standard MAC interfaces do not receive what they are transmitting – each site behaves as if it misses own messages, which reduces the benefits of the ContiguityDetector and may lead to the graph  $G$  growing too large to be processed in real time; we have tried a Suspended Delivery option whereby upon multicast of a message  $Z$ ,  $X \Rightarrow Y$  is inferred for all subsequently received messages  $X, Y, \dots$  (disregarding the apparent miss of message  $Z$ ), whose delivery is however suspended until reception of a *report* PDU with a graph  $G'$  containing vertex  $Z$ .

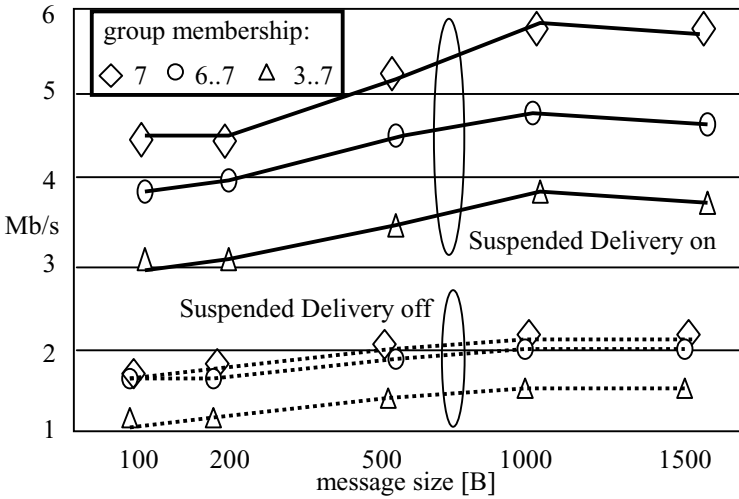


Fig. 3 Group throughput measurements for the DPG protocol

Measurements for a 7-site group reveal the significance of the latter option (Fig. 3). With it, the graphs  $G$  at sites rarely exceed 30 vertices and the group throughput is practically limited only by the PCs' processing rates. For example, over 740 1000-byte messages are delivered in ADO per second, with a graceful drop under variable membership, reflecting only the diminishing contribution of the fewer sites present, and not the reconfiguration overhead typical of existing ADO protocols [1], [6], [14], [15]. Switching off the Suspended Delivery option was not unlike switching off the ContiguityDetector altogether and typically resulted in reducing the group throughput to one-third of the previous value or less.

## 7 Conclusion

GCPs designed to serve collective output-oriented applications should be concerned about ADO, but much less about Membership Control. Under variable group membership, such protocols should ideally exhibit a group throughput drop that reflects only the diminishing contribution of the fewer member sites. We have proposed a membership-insensitive quasi-ADO protocol, built on top of the 1C

network service complete with a conceptual inference device called ContiguityDetector, tailored to communication environments where message sequencing comes as a 'natural' feature e.g., single-access LANs, bounded-delay or tag-sequencing WANs. The proposed protocol has been implemented in a 10 Mb/s LAN environment to verify that it is able to maintain a high group throughput of messages delivered in quasi-ADO. A challenging issue is running the protocol in a larger group (50 or more sites); at the moment this is only possible via simulation.

## References

1. Amir Y., Moser, L.E., Melliar-Smith, P.M., Agarwal, D.A., Ciarfella, P.: Fast Message Ordering and Membership Using a Logical Token-Passing Ring. In: Proc. 14th Int. Conf. on Distributed Computing Systems (1993) 551-560
2. Birman, K.P., Schiper, A., Stephenson, P.: Lightweight Causal and Atomic Group Multicast. ACM Trans. on Computer Systems 9 (1991) 272-314
3. Bruck, J., Dolev, D., Ho, C., Orni, R., Strong, R.: PCODE: An Efficient and Reliable Collective Communication Protocol for Unreliable Broadcast Domains. <ftp://cs.huji.ac.il/pub/TRANSIS> (1994)
4. Cheriton, D.R., Skeen, D.: Understanding the Limitations of Causally and Totally Ordered Communication. In: Proc. 14th ACM Symp. on Operating System Principles (1993) 44-57
5. Diot, C., Dabbous, W., Crowcroft, J.: Multipoint Communication: A Survey of Protocols, Functions and Mechanisms. IEEE J. on Selected Areas in Comm. 15 (1997) 277-290
6. Dolev D., Kramer, S., Malki, D.: Early Delivery Totally Ordered Multicast in Asynchronous Environments. In: Proc. 23rd Int. Symp. on Fault-Tolerant Computing (1993) 544-553
7. Ezhilchelvan, P.D., Macedo, R.A., Shrivastava, S.K.: Newtop: A Fault-Tolerant Group Communication Protocol. In: Proc. 15th Int. Conf. on Distributed Computing Systems (1995) 336-356
8. Fischer, M., Lynch, N.A., Paterson, M.S.: Impossibility of Distributed Consensus with One Faulty Process. J. of the ACM 32 (1985) 374-382
9. Hadzilacos, V., Toueg, S.: Fault-Tolerant Broadcasts and Related Problems. In: Mullender, S. (ed.): Distributed Systems, Addison-Wesley, Wokingam Reading (1993) 97-145
10. Melliar-Smith, P.M., Moser, L.E., Agrawala, V.: Broadcast Protocols for Distributed Systems. IEEE Trans. on Parallel and Distributed Systems 1 (1990) 17-25
11. Moser L.E., Amir, Y., Melliar-Smith, P.M., Agarwal, D.A.: Extended Virtual Synchrony. In: Proc. 14th Int. Conf. on Distributed Computing Systems (1994) 56-65
12. Nakamura, A., Takizawa, M.: Priority-Based and Semi-total Ordering Broadcast Protocols. In: Proc. 12th Int. Conf. on Distributed Computing Systems (1992) 178-185
13. Obraczka, K.: Multicast Transport Protocols: A Survey and Taxonomy. IEEE Comm. Magazine 36 (1998) 94-102
14. Rodrigues, L., Fonseca, H., Verissimo, P.: Totally Ordered Multicast in Large-Scale Systems. In: Proc. 16th Int. Conf. on Distributed Computing Systems (1996) 503-510.
15. Whetten, B., Montgomery, T., Kaplan, S.: A High-Performance Totally Ordered Multicast Protocol. In: Theory and Practice in Distributed Systems, Lecture Notes in Computer Science, Vol. 938, Springer-Verlag, Berlin Heidelberg New York (1994)
16. Wilhelm, U., Schiper, A.: A Hierarchy of Totally Ordered Multicasts. <http://www-uk.research.ec.org/broadcast/trs/papers/85.ps>

# On Finding Feasible Solutions to the Group Multicast Routing Problem

Ning Wang and Chor Ping Low

School of Electrical & Electronic Engineering  
Nanyang Technological University  
Singapore 639798  
Republic of Singapore  
{P145139905, icplow}@ntu.edu.sg

**Abstract.** Group multicast routing problem (GMRP) is a generalization of multicasting whereby every member of the group is allowed to multicast messages to other members from the same group. The routing problem in this case involves the construction of a set of low cost multicast trees with bandwidth requirements for all the group members in the network. The traditional solutions only care for the low cost of multicast trees and sometimes the algorithms will fail during the construction due to the inefficiency of the bandwidth allocation. In this paper we study the feasible solutions to GMRP by proposing a new algorithm to improve the success rate of constructing multicast trees. Simulation results show that our new algorithm performed better in terms of bandwidth utilization and success rate of building multicast trees compared with existing algorithms.

## 1 Introduction

Nowadays most of the network applications require large amount of bandwidth to deliver multimedia information to multiple destinations simultaneously. One possible way to meet this requirement is via multicast communication. Multicasting allows a source to send information to multiple destinations through a network at the same time. Multicast routing can be solved by building multicast trees over the network topology, and transmit data from the source to all the destinations. A least cost multicast tree is referred to as a *Steiner tree*. The problem of finding a Steiner tree has been proved to be NP-complete by [4].

Group Multicast Routing Problem (GMRP) is a generalization of multicast routing whereby each member node from a group may multicast data to all other members from the same group, i.e. each member node being both an information source and destination. A typical example is that of remote teleconferencing in which every participant is able to send out messages and receive information presented by other group participants concurrently. It can be easily inferred that group multicasting will need higher bandwidth resources than the corresponding “single source” multicast routing.

One possible solution to group multicast routing problem is to use Core Based Tree (CBT) technology [1] with only one shared tree generated, rooting at the “multicasting core” and spans to all the group member nodes with minimal cost. However, using CBT will result in side effects such as network congestion on some tree edges connected to the “core” because data from each source will have to traverse these edges. Too large delay variation is another problem of using CBT algorithm. An alternative method is to create one Steiner tree for each source separately, as [5] and [9] have proposed. However, one common deficiency of the two algorithms lies in the unbalanced data concentration throughout the network. Edges with low cost are always burdened with high traffic loading while other edges are seldom utilized. Extremely, since the two greedy algorithms always select low cost edges to build multicast trees, there exists the situation that the bandwidth of some critical edges are used up and the network becomes disconnected, thus resulting in the failure of the tree construction.

In this paper, we propose a new routing algorithm for group multicast routing problem called Feasible Solutions using adapted TM algorithm for GMRP (FTM). Our main goal is to provide better distribution of data in the network and more important, lower tree failure rate. The new heuristic is based on the adaptation of the *TM algorithm* by H. Takahashi and A. Matsuyama [11]. Extensive simulations are carried out to compare the performance of our proposed algorithm with the traditional algorithms concerning the tree failure rate, variance of traffic loading and the edge saturation which can be used to demonstrate data distribution in the network.

## 2 Problem Formulation

Generally the network is modeled as a directed graph  $G(V, E)$  with node set  $V$  ( $|V|=n$ ) and edge set  $E$ . Each edge  $(i, j) \in E$  has two parameters, namely available bandwidth  $b_{ij}$  and cost  $c_{ij}$ . We assume that the available bandwidth on each edge is asymmetric in general, i.e.  $b_{ij}$  may not be equal to  $b_{ji}$ . For each edge  $(i, j)$ ,  $b_{ij}$  is known as the *input bandwidth* from node  $i$  to node  $j$ . For simplicity, we assume that the cost of each edge  $(i, j)$  is symmetric, i.e.  $c_{ij} = c_{ji}$ . However, our proposed algorithm can be easily adapted to the case in which the cost of each edge is asymmetric. Let  $P(u, v)$  denote the shortest path (in terms of cost) from  $u$  to  $v$ , where  $u, v \in V$ .

Given a network graph  $G=(V, E)$ , let  $D = \{v_1, v_2, \dots, v_m\}$  be a group of nodes in  $G$ , where  $D \subset V$  and  $|D|=m$ . Each node  $m_k \in D$  has a bandwidth requirements of  $B_k$  units. The group  $D$  is called the *multicasting group* and each node in  $D$  is called a *member node*. The bandwidth requirement  $B_k$  of each node in  $D$  is specified by the user. The *group multicast routing problem (GMRP)* is that of finding a set of directed routing trees  $\{T_1, T_2, \dots, T_m\}$ , one for each member of group  $D$  which satisfy the following requirements:

$$\text{minimize} \quad \sum_{k=1}^m \sum_{i,j \in T_k} c_{ij} X_{ij}^k, \quad i, j \in V \quad (1)$$

$$\sum_{k=1}^m B_k X_{ij}^k \leq b_{ij} \quad \text{where} \quad X_{ij}^k = \begin{cases} 1 & \text{if } (i, j) \in E_k \\ 0 & \text{otherwise} \end{cases} \quad (2)$$

Each tree  $T_i = (V_i, E_i)$ , where  $V_i \subseteq V$  and  $E_i \subset E$ , represents the tree rooted at node  $v_i \in D$  which spans all nodes in  $D$ . The tree  $T_i$  may also contains some nodes from the set  $V-D$ , which we call the *relay nodes*. We note here that all leaves nodes in each tree  $T_i$  must be member nodes of  $D$ . Otherwise, the relay nodes and their corresponding links can be deleted from the tree resulting in a lower cost solution.

The first constraint ensures that the total cost of generated trees is minimized while the second constraint is to ensure that the total bandwidth utilized on each link does not exceed its available bandwidth. Since GMRP is the generalization of single Steiner tree problem, it is obviously NP-complete. A set of trees  $\{T_1, T_2, \dots, T_m\}$  which satisfy constraint (2) is called a *feasible solution* to the group multicast routing problem, i.e., our goal is to achieve the maximum success rate of building multicast trees. In this paper we assume that when each edge is joined into the tree, one unit of its bandwidth will be consumed. An edge is said to be *saturated* if its consumed bandwidth has reached the bandwidth capacity of this edge.

Fig. 1 shows a network model as we have defined. The number in the middle of the edge is cost and the numbers near the arrows at the two ends of an edge represent the bandwidth capacity of the edge in the direction of the arrows. Note that the nodes in gray color are relay nodes and the rest are multicasting members in the network, i.e.,  $D = \{0, 1, 2, 3\}$ .

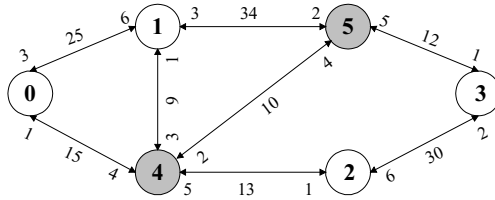


Fig. 1 Network model

### 3 Previous Works

Single source based multicast routing with minimum cost is referred to as the Steiner tree problem. L. Kou, G. Markowsky and L. Berman proposed a heuristic algorithm called KMB algorithm to achieve the close-to-optimal solutions [8]. The KMB algorithm begins by computing shortest paths between each pair of member nodes. Following that, the closure graph  $G'$  which contains only nodes in  $D$  is constructed. The closure graph  $G'$  is one in which each node in  $D$  is connected to all other nodes in  $D$  by the shortest paths. Thus, each edge  $(a, b)$  in the closure graph correspond to a shortest path between nodes  $a$  and  $b$  in the original graph  $G$ . The next step is to construct minimum spanning tree of  $G'$ . A greedy approach is employed to add edges

to the partial tree until all member nodes in  $D$  are included. In each iteration of this approach, a least cost edge that connects a node in the partial tree to one that is not in the partial tree is selected to be joined. Once the tree is constructed, all edges in the tree are expanded into the shortest paths which they represent in the original network.

H. Takahashi and A. Matsuyama proposed TM algorithm [11] by computing the shortest paths from all the nodes in the network to every group members. Empirical study [9] showed that TM algorithm performs better than KMB according to total cost due to its more powerful capacity of explore shortest paths. When one member node is about to join into the Steiner tree, not only the shortest paths from the member nodes already in the tree but also those from some relay nodes can be obtained to build the Steiner tree.

According to the problem of group multicast routing, multicast tree will be built on each of the member nodes and the bandwidth requirement is taken into consideration. Jia and Wang [5] first proposed a heuristic algorithm, which we call *Jia & Wang's Algorithm*, which adopts some form of coordinated strategy called *overhead comparison* to generate a set of multicast trees. Their algorithm is based on an adaptation of KMB Steiner tree heuristic. Later on another heuristic called Group multicast routing with adapted TM algorithm (GTM, [9]) was proposed. As we have mentioned above, due to the more powerful capacity of TM algorithm to explore shortest paths, GTM correspondingly provides better performance in total cost than *Jia & Wang's algorithm*. Moreover, GTM also obtains lower tree failure rate compared with *Jia & Wang's algorithm* [9].

It should be noted that the problem of finding multicast tree with minimum cost is NP-complete [4], being a generalization of this problem, none of the solutions to GMRP we have mentioned can find the optimal solution in polynomial time. We have also found that even if there exist some solutions to building multicast tree for each group member in the network, cases are that neither *Jia & Wang's algorithm* nor GTM can find any. In this paper, we will mainly focus on how to achieve the maximum success rate of building a set of multicast trees, i.e., if there do exist solutions to the problem, our proposed algorithm can always find out at least one that is feasible.

## 4 Proposed Algorithm

### 4.1 Breadth First Search (BFS) Tree

Observe GTM and other similar algorithms which adopt greedy strategy to build multicast trees, edges with low cost are in higher priority to be selected each time so that those edges also having low bandwidth capacity could be consumed up and become saturated before all the multicast trees are generated. If unfortunately some critical edges are unavailable in the network, all the group members are not located in one connected sub-graph, the algorithm fails. One sufficient condition to prevent the network from being disconnected is to guarantee that there are no saturated edges in the network. Based on the above analysis, FTM always finds edges with large bandwidth capacity to build multicast trees. Following this scenario, we first find the “widest” path from all the nodes to every group members, as TM algorithm deals with

cost in the network. To achieve this, we use breadth first search (BFS) to detect the paths with maximum bandwidth capacity to the group members. One BFS tree is built rooted at each multicasting group member and contains the widest path from all the other nodes in the graph. When each of the nodes is visited during BFS procedure, the edges from all its neighbors (except its predecessor) to the node itself are examined, and the one with maximum bandwidth capacity is selected, with the corresponding neighbor included into the search tree. At the same time, the bandwidth bottleneck and the total cost of the path to the root is also recorded. If there are two or more neighbors connected by the edges with the same bandwidth capacity, the one with least cost is chosen. In order to implement the breadth first search, a queue is used to accommodate the nodes to be visited. Fig. 2 shows how the BFS tree is generated for node 0 in the graph given by Fig. 1. Notice that the two numbers in the bracket represent the total cost and bandwidth bottleneck of the path from this node to node 0 respectively.

Starting from node 0, since the two of its neighbors node 1 and node 4 have never been visited, they are joined into the BFS tree with bandwidth bottleneck 3 and 1 respectively as Fig 2(a) shows. Next we come to examine node 1's neighbors. Node 5 is joined into the BFS tree by connecting node 1, its bottleneck to the source is  $\min(\text{bottleneck}[1], b(5,1))$ , which is also 3. As to node 4,  $\min(\text{bottleneck}[1], b(4,1))$  is 1, the same with  $\text{bottleneck}[4]$ , however the total cost of path  $4 \rightarrow 1 \rightarrow 0$  is 34 which is greater than the cost of edge  $(4,1)$ , so node 4 is remained directly connecting to node 0. After all of node 1's neighbors have been visited, we come to examine node 4's neighbors. Since the bottleneck of both node 1 and node 5 are greater than that of node 4, they remain unchanged. As to node 2, since it has never been visited, it is joined into the tree by connecting to node 4 (Fig. 2(b)). Then we will come to check node 5's neighbors. Node 3 will be added into the BFS tree with its bottleneck being  $\min(\text{bottleneck}[5], b(3,5))=3$  (Fig. 2(c)). Since  $\min(\text{bottleneck}[5], b(4,5))=3$  is greater than  $\text{bottleneck}[4]$ , node 4 will switch its path from node 0 to node 5, also notice that the path cost and bottleneck of its current successor node 2 will also be updated to 82 and 3 respectively, as Fig. 2(d) illustrates. Next the neighbors of node 2 are visited. Since the path  $3 \rightarrow 2 \rightarrow 4 \rightarrow 5 \rightarrow 1 \rightarrow 0$  and  $3 \rightarrow 5 \rightarrow 1 \rightarrow 0$  have the same bandwidth bottleneck while the cost of the latter is smaller, node 3 will not switch its connection from node 5 to node 2. Finally the neighbors of 3 are checked and we can find that there is no need to modify any connections. After the breadth first search is finished, Fig. 2(d) is the final BFS tree for node 0 with the records of the bandwidth bottleneck and total cost of the paths from all the other nodes to the root.

## 4.2 Necessary Conditions for Existence of Feasible Solutions.

While the problem of finding feasible solutions for GMRP is NP-complete, there exists some criteria which could be used to determine if a feasible solution does exist for a given problem instance. Two of such criteria are given in the following lemma. Here we still assume that one unit of bandwidth is consumed when each edge is included in the multicast tree.

**Lemma:** There is no feasible solution for GMRP if :

- (i) all member nodes do not belong to the same connected component; or
- (ii) the total input bandwidth capacity for some member node  $v$  is less than  $m-1$ .



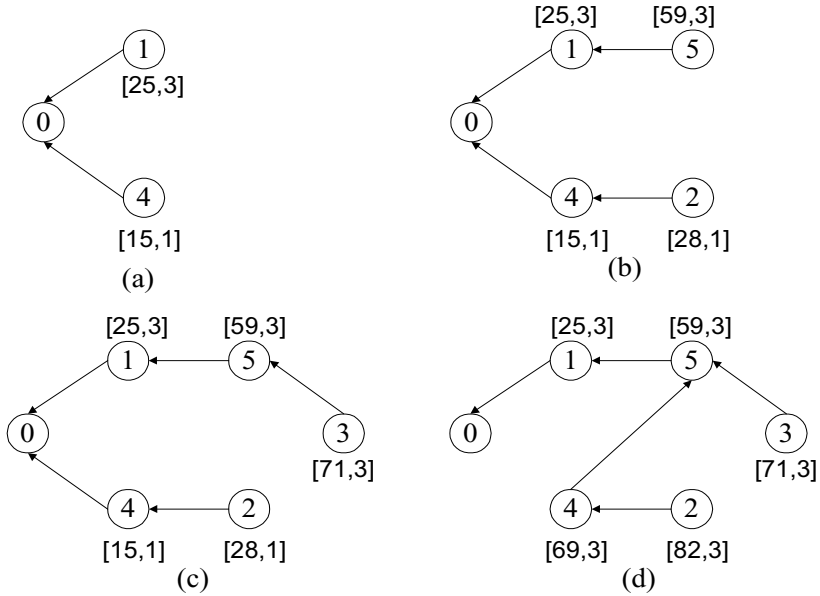


Fig. 2 Generation of BFS tree rooted at node 0

**Proof:** A feasible solution for GMRP is comprised of a set of multicast trees  $\{T_u : u \in D\}$ . Each tree  $T_u$  is rooted at member  $u \in D$  and spans all other member nodes in the network. If member nodes are located in two or more components, then it would not be possible to construct a multicast tree that connects one member node to all other member nodes in the network. It follows then that criteria (i) is a necessary condition for the existence of feasible solutions for GMRP.

In any feasible solution, each member node  $v$  will be receiving information from the other  $m-1$  member nodes via  $m-1$  multicast trees  $T_u$ , for each  $u \in D - \{v\}$ . Thus, the total input bandwidth of each member node must be at least  $m-1$ .

This lemma can be used as an *early-abort test* to determine in advance if a feasible solution exists before attempting to find any solutions for GMRP.

### 4.3 Feasible Solution to GMRP

After  $m$  BFS trees are built rooted at each of the group members, the detail information of the widest path from all the other nodes to this group member is available. Similar to the TM algorithm, our next step is to build multicast trees for the group members based on the path information obtained by BFS procedure. It should be noted that bandwidth is a concave metric in the network model, i.e., the bandwidth capacity of a path is decided by the edge with minimum available bandwidth along this path. Since the bandwidth bottleneck of the path is recorded in BFS procedure, we will use the path with maximum bandwidth bottleneck to join all the group

members into the current multicast tree. Different from TM algorithm, in this step paths are selected according to the bandwidth bottleneck instead of the edge cost. Of course, if two or more paths with the same bandwidth bottleneck are found at the same time, the one with least cost is selected. Fig. 3 gives out the pseudocode of the feasible solution to GMRP.

---

**Procedure BFS:**

**Input:** graph  $G(V, E)$ , source;

**Output:** Maximum bandwidth path to source;

**begin**

**for**  $i=1$  **to**  $n$  **do** /\* Initialization \*/

**if** (  $i < \text{source}$  ) **then**

$\text{node}[i].\text{visited}=0$ ;

$\text{distance}[i]=\text{infinite}$ ;  $\text{bottleneck}[i]=-1$ ;  $\text{predecessor}[i]=-1$ ;

**else**

$\text{distance}[i]=0$ ;  $\text{bottleneck}[i]=\text{infinite}$ ;  $\text{predecessor}[i]=-1$ ;

**end for**;

$T=\emptyset$ ;

    InitQueue(Q);

**for**  $i=1$  **to**  $n$  **do**

**if** (  $\text{node}[i].\text{visited}=0$  ) **then**

            EnQueue(Q, i);

**while** ( **not** QueueEmpty(Q) ) **do**

$u = \text{DeQueue}(Q)$ ;

$u.\text{visited}=1$ ;

**for each**  $v$  **such that**  $(v, u) \in E$  **do** /\* Find the neighbor with

**if** (  $\min(\text{bandwidth}(v, u), \text{bottleneck}[u]) > \text{bottleneck}[v]$  ) **or** maximum bandwidth edge \*/

                    (  $\min(\text{bandwidth}(v, u), \text{bottleneck}[u]) = \text{bottleneck}[v]$  ) **and**

                    (  $\text{distance}[v] > \text{distance}[u] + \text{cost}(v, u)$  ) **then**

$(v, u) \rightarrow T$ ;

$\text{bottleneck}[v] = \min(\text{bandwidth}(v, u), \text{bottleneck}[u])$ ;

$\text{predecessor}[v] = u$ ;

$\text{distance}[v] = \text{distance}[u] + \text{cost}(v, u)$ ;

**for each**  $w$  **such that**  $(v, w) \in T$  **do**

                            update  $w$ 's bottleneck and distance;

**end if**;

**for each**  $v$  **such that**  $(v, u) \in E$  **do**

**if**  $\text{node}[v].\text{visited}=0$  **then**

                    EnQueue(Q, v);

**end while**

**end if**;

**end**; (Procedure BFS)

**Procedure FTM**

**Input:**  $G(V, E)$ ,  $D$ ;

**Output:** A set of multicast trees  $T_v$  for each  $v \in D$ .

**begin**

**if**  $G$  is not connected **then** stop;

**for each** member node  $v \in D$  **do** /\* early-abort detection \*/

**if** total input bandwidth for  $v < m-1$  **then** stop;

**for each** member  $i \in D$  **do**

**for each** member  $i \in D$  **do** BFS( $G, i$ );

            set  $T_i = (V_i, E_i)$ ,  $V_i = \{i\}$ ,  $E_i = \emptyset$ ,  $\text{copies}=1$ ;

```

while (copies < m)
    find a path  $P(v, u)$  where  $v \in V_i, u \in D - V_i$ , such that /* Select the path with
     $bottleneck(v, u) = \max_{i \in V^*, j \in D - V^*} bottleneck(i, j)$  maximum bandwidth bottleneck */
    if there exist more than one  $P(v, u)$  with the same bottleneck then
        select the path with least cost;
     $V_i = V_i \cup \{\text{all nodes in } P(v, u) \text{ except } v\}$ ;
     $E_i = E_i \cup \{\text{all edges in } P(v, u)\}$ ;
    copies = copies + 1;
end while
update bandwidth status;
end for
end; (Procedure FTM)

```

---

**Fig. 3** Psudocode of feasible solution to GMRP

#### 4.4 Time Complexity Analysis

In the function of BFS, it takes  $O(n+e)$  to visit each node in the graph using breadth-first-search where  $e$  is the number of edges in the graph. The time complexity of visiting all the neighbors of each node is  $O(n)$  while updating the distance and bandwidth bottleneck of successors takes  $O(n)$ . Hence the time complexity of BFS is  $O((n+e)n^2)$ . In the function of FTM, BFS dominates most of the computing time when building multicast trees for each group member. On the whole, building  $m$  multicast trees will take  $O(m(n+e)n^2)$  which is actually the time complexity of our proposed FTM algorithm.

#### 4.5 An Example

We also take the network model shown in Fig. 1 as our example. Since node 0,1,2,3 are multicasting group members, one multicast tree will be built for each of them respectively. First we use the traditional greedy method to solve the problem. Fig. 4(a) shows the Steiner tree rooted at node 0 with optimal total cost. However, after tree 0 is finished, we find that edge (4, 2) and (5, 3) have been saturated, resulting in the separation of node 2, 3 from the rest part of the graph, as shown in Fig. 4(b). Definitely there will be no feasible solutions to building the rest of the multicast trees due to the fact that the network has been disjointed.

In our proposed algorithm, the multicast trees are built based on the path information provided by the BFS trees rooted at each group member. In order to generate each multicast trees, two tables must be available: the table of bandwidth bottleneck and the table of path cost, both of the tables are  $m*n$  in size. Fig. 5 shows the two tables needed to build multicast tree 0. In the first table, element  $(i, j)$  represents the bandwidth bottleneck from node  $(j)$  to group member  $(i)$  and in the other, element  $(i, j)$  represents the total cost of the path from node  $(j)$  to group member  $(i)$ . Notice that the data in the  $i$ th row of both tables are obtained by the BFS

tree rooted at node  $i$ . Starting from node 0, we check the first column of table 1 and find that among the bottlenecks of all the nodes, node 1 has the widest path to node 0 which is 6, so it is selected into the multicast tree rooted at node 0. After node 1 is joined into the multicast tree, we find that both node 2 and node 3 have the bandwidth bottleneck of 1 from the partially built tree which currently contains node 0 and node 1 (shown as the left-down 2 by 2 elements in table 1), so table 2 is checked and node 2 is selected into the tree by taking the path  $1 \rightarrow 4 \rightarrow 2$  whose cost is 22 which is less than all the other possible paths from node 3 (58 from node 0 and 46 from node 1). Finally node 3 is added into the multicast tree by taking the path  $2 \rightarrow 3$ , as shown in Fig. 6(a). Similarly the multicast trees rooted at the rest of the members are built as Fig. 6(b),(c),(d) shows.

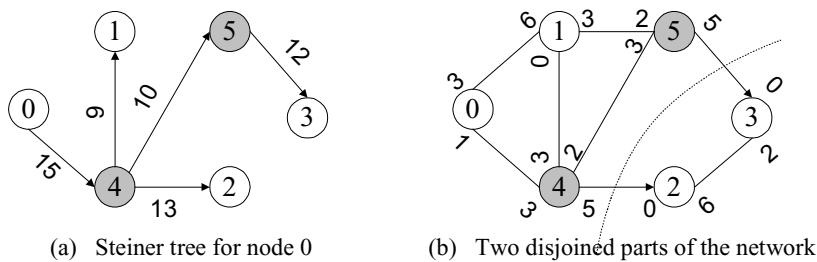


Fig. 4 Traditional algorithms for GMRP

	0	1	2	3	4	5
0	--	3	3	3	3	3
1	6	--	3	3	3	3
2	1	1	--	6	1	1
3	1	1	2	--	1	1

(1) Table of bandwidth bottleneck

	0	1	2	3	4	5
0	--	25	82	71	69	59
1	25	--	57	46	44	34
2	28	22	--	30	13	23
3	58	46	30	--	43	12

(2) Table of path cost

Fig. 5 Two tables for building multicast tree 0

5 Simulations

5.1 Random Graph Generation

To ensure that simulation of the effects of different routing algorithms are fairly evaluated, random graphs with low average degrees, which better represent the topologies of common point-to-point network, e.g. NSFNET, are constructed. The nodes are randomly placed on a rectangular grid and nodes are connected with the probability function:

$$P(u,v) = \lambda \exp(\frac{-d(u,v)}{\rho L})$$

where  $d(u,v)$  is the distance between node  $u$  and  $v$  and  $L$  is the maximum possible distance between any pair of nodes. The parameters  $\lambda$  and  $\rho$  ranging  $(0,1]$  can be modified to create the desired network model. For example, a large value for  $\lambda$  gives nodes with a high average degree, and a small value for  $\rho$  increases the density of shorter links relative to longer ones. In our simulation,  $\lambda$  and  $\rho$  are set to 0.4 and 0.2, respectively. We use the distance between node  $u$  and  $v$  as the cost of the edge. The bandwidth capacity of each edge is allocated using the following function:

$$B(u,v) = B_m + (-1)^k \times r \bmod B_m$$

where  $B_m$  is the given mean bandwidth while  $k$  and  $r$  are random numbers. Using this function no bandwidth with negative value will be generated and the bandwidth capacity of all edges will range from 1 to  $2B_m - 1$ . Graphs are generated and tested until member nodes are connected in a single sub-graph.

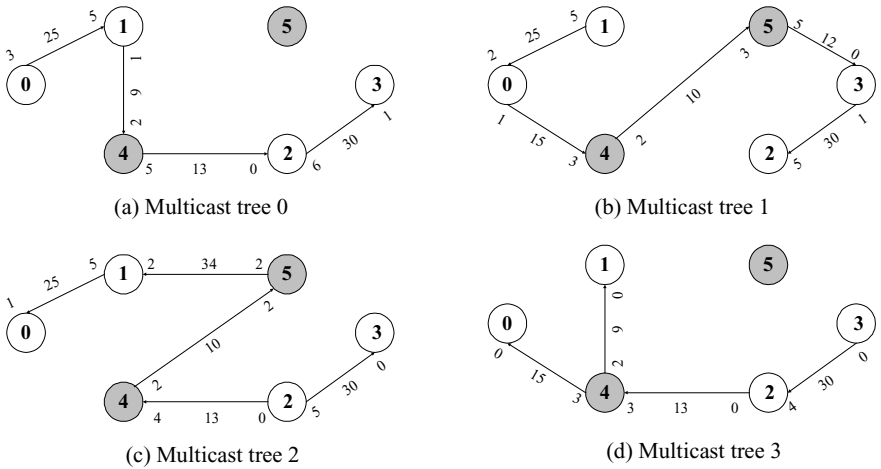


Fig. 6 A feasible solution

## 5.2 Algorithm Performance Study

In our simulation, we will mainly compare the performance between the proposed FTM algorithm and such existing heuristics as *Jia & Wang's algorithm* and GTM.

First we consider the data distribution in the network. We define link loading factor  $L$  for each edge:

$$L_{ij} = \frac{\sum_{k=1}^m B_k X_{ij}^k}{b_{ij}}$$

and network loading factor:

$$\bar{L} = \frac{\sum_{(i,j) \in E} L_{ij}}{N_e}$$

where  $N_e$  is the number of edges in the network. From definition, we can see that  $L_{ij}$  is actually the rate between the consumed bandwidth and the bandwidth capacity of this edge.  $\bar{L}$  is the average loading of all edges in the network. If the variations of  $L_{ij}$  between each edge are small, we can draw the conclusion that the data traffic is well distributed in the network.

Fig. 7 shows the network loading factor over the group size. The network size is fixed at 100 and the mean bandwidth is fixed at 30. From Fig. 7 we can see that the network loading factor increases as the group size grows larger. This phenomenon is expected because with the increase of the group size, not only more multicast trees will be built, but the size of each tree will become larger. We also find that the network loading factor of FTM is higher than that of *Jia & Wang's algorithm* and GTM by a small scale.

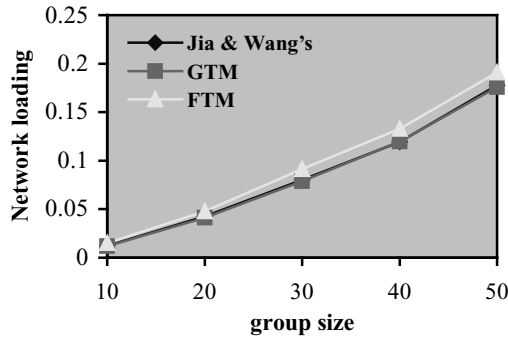


Fig. 7 Traffic loading factor vs. group size

Since the network loading factor is comparable between the two algorithms, we will continue to study the data distribution in the network by computing the variance of loading factor  $\delta$ . We define:

$$\delta = \sum_{(i,j) \in E} P_{ij} \times (L_{ij} - \bar{L})^2$$

where  $P_{ij}$  is the probability that edge  $(i,j)$  is selected into the multicast trees.  $P_{ij}$  can be expressed as the number of times that edge  $(i,j)$  is selected into the multicast trees over the total times that all edges are selected. Since one unit of bandwidth is consumed when each edge is joined, we can express the probability with the consumed bandwidth as the following:

$$P_{ij} = \frac{\sum_{k=1}^m B_k X_{ij}^k}{\sum_{(u,v) \in E} \sum_{k=1}^m B_k X_{uv}^k}$$

Fig. 8 shows the variance of loading factor over group size. Notice that all the external conditions are same as those of Fig. 7. From the figure we can see that the variance of loading factor of FTM is significantly lower than that of the other two

algorithms, typically when the group size is 50, variance of FTM is lower than that of GTM by 67.9% and *Jia & Wang's algorithm* by 69.2%. Since the traffic loading are very close to each other (see Fig. 7), low traffic loading variance reflects better distribution of data in the network. In FTM, the total traffic is evenly allocated to a large number of network links and each of them has a moderate traffic loading. On the contrary, edges with low cost are burdened with high traffic loading while those with high cost are seldom utilized in *Jia & Wang's algorithm* and GTM, resulting in the unbalanced data distribution.

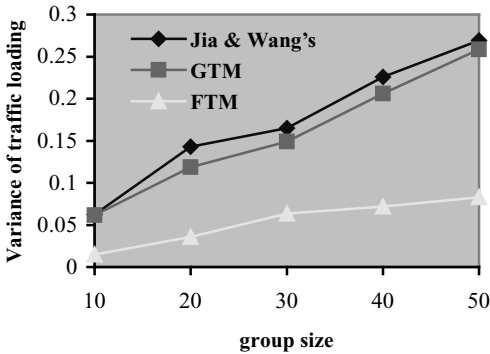


Fig. 8 Variance of traffic loading vs. group size

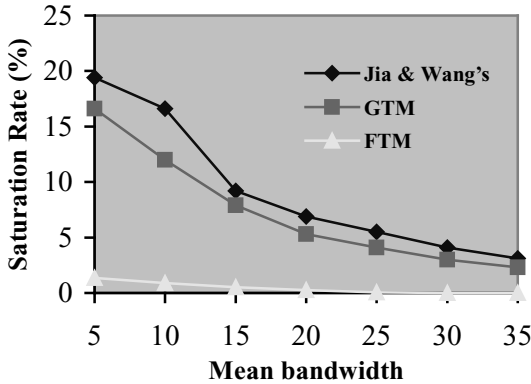


Fig. 9 Saturation rate vs. mean bandwidth

Next we will study the relationship between mean bandwidth and edge saturation rate. Edge saturation rate is defined as the number of saturated edges over the number of all edges in the network after the algorithm terminates. Fig. 9 shows the saturation rate over the mean bandwidth. The network size is fixed at 100 and the number of group members is 30 while the mean bandwidth ranges from 5 to 35 in steps of 5. From the figure we can see that all of the three algorithms have fewer saturated edges as the mean bandwidth increases. However, the saturation rate of FTM is obviously lower than the other two, e.g. when the given mean bandwidth is as low as 5, the edge

saturation rate of GTM reaches 17% while that of FTM is only 1.35% under this condition. The phenomenon can be explained by the fact that the edges with lower bandwidth capacity have lower priority to be selected to build the multicast trees so that the bandwidth is conserved. By studying the edge saturation condition, we can also infer that FTM can provide better distribution of data in the network. Since the edges with large bandwidth capacity is selected each time, their priority decreases as the bandwidth is consumed step by step while the priority of other edges will become higher. Following this scenario, a dynamic balance among the edge selection is achieved and data can be more evenly distributed in the network.

Fig. 10 shows the tree failure rate over the mean bandwidth. Similarly, the network size is also fixed at 100 and the number of group members is 30 while the mean bandwidth ranges from 5 to 35 in steps of 5. Observe that tree failure rate of both algorithms converges to zero as the mean bandwidth increase. This phenomenon is expected as the chance of finding paths with sufficient bandwidth capacity increases as the mean bandwidth increases. From Fig. 10 we can also see that the tree failure rate of FTM is significantly lower than that of *Jia & Wang's algorithm* and GTM. The reason for this is due to the low edge saturation rate as we have illustrated above. When the given mean bandwidth is high, each of the edges is not easy to be saturated. In this case, as long as all the group members are located in one connected graph in the initial condition which we will examine before each run, the topology of the graph won't change if no edges are saturated and there must always exist solutions.

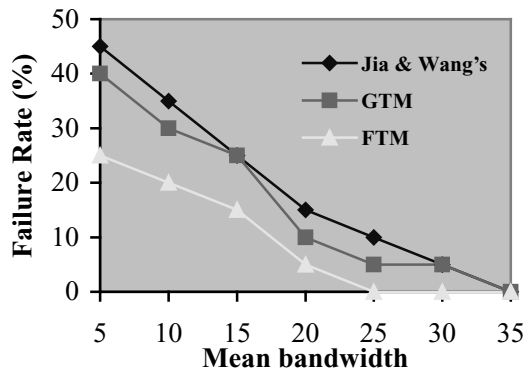
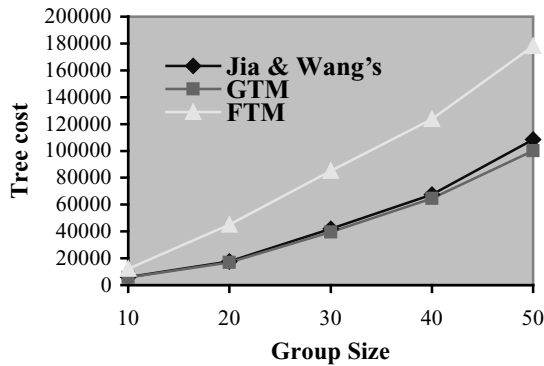


Fig. 10 Tree failure rate vs. mean bandwidth

As we have mentioned in Section 2, our proposed algorithm focuses on the feasible solution to GMRP, and the factor of cost is not our main concern. As the algorithm proceeds, only if two or more paths having the same bandwidth bottleneck are met will we consider cost efficiency and choose one path with least cost. Fig. 11 shows the performance according to total tree cost vs. group size. From the figure we can find that FTM has significantly higher cost than the other two greedy heuristics, typically when the group size reaches 50, the total cost of *Jia & Wang's algorithm* and GTM are 60.9% and 56.2% that of FTM respectively. How to provide efficient tradeoff between total tree cost and failure rate is one of the topics that our future works will focus on.





**Fig. 11** Total tree cost vs. group size

## 6 Conclusions

In this paper, we examine feasible solutions to group multicast routing problem with bandwidth requirements. In order to provide better multicasting data distribution and lower tree failure rate, we proposed a new algorithm called feasible solution to GMRP using adapted TM algorithm (FTM). Results from our empirical study shows that our new algorithm performs better in terms of data distribution and edge saturation rate compared to other greedy algorithms. In addition, our simulations also show that FTM has a higher percentage of success in finding solutions to GMRP due to the efficient edge selecting strategy.

Currently our proposed algorithm mainly concerns on the success rate of building multicast trees in the network. Cost does not contribute much to the edge selections so that the resulting trees are of higher cost than other greedy algorithms. In addition, the proposed algorithm is based on a centralized fashion, i.e., the network topology is assumed to be obtained by all the nodes in the graph, and we will later move on to the distributed fashioned algorithms for GMRP which will be more feasible for “actual” networks. One of our future research directions will also focus on the network model with guaranteed QoS and provide trade-off between success rate and tree cost.

## References

- [1]. Ballardie T, Francis P., Crowcroft J., “*Core Based Trees (CBT)*”, Proc. SIGCOMM '93 Conf., ACM, (1993) 85-95.
- [2]. F. Bauer. A. Varma. “*Distributed algorithms for multicast path setup in data networks*”, IEEE/ACM Trans. Networking 4(2), (1996) 181-191.
- [3]. Dirceu Cavendish, Aiguo Fei, Mario Gerla and Raphael Rom, “*On the construction of low cost Multicast Trees with Bandwidth Reservation*”, UCLA Technical Report, Doc ID: ncstrl.ucla\_cs/970043.

- [4]. M.R. Garey, D.S. Johnson, "*Computers and Intractability—a Guide to the Theory of NP-completeness*", Freeman, New York, 1979.
- [5]. X. Jia and L. Wang, "*Group multicast routing algorithm by using multiple minimum Steiner trees*", Computer Communications 20, (1997) 750-758.
- [6]. R. M. Karp, "Reducibility among Combinatorial Problems" in Miller and Thatcher (Eds.), *complexity of Computer Computations*, Plenum Press. New York, 1972, 85-103.
- [7]. V. P. Kompella, J. C. Pasquale, G. C. Polyzos. "*Multicast routing for multimedia communication*", IEEE/ACM Tran. On Networking 1(3), (June 1993) 286-292.
- [8]. L. Kou, G. Markowsky and L. Berman, "*A fast algorithm for Steiner trees*", Acta Informatica 15, (1981) 141-145.
- [9]. C. P. Low, N. Wang "*An efficient algorithm for group multicast routing with bandwidth reservation*", Proc. IEEE International conf. on networks (ICON'99).
- [10]. V.J. Rayward-Smith and A. Clare, "On Finding Steiner Vertices", Networks, Vol. 16, no. 3, (1986), 283-294.
- [11]. H. Takahashi and A. Matsuyama, "*An approximate solution for the Steiner problem in graphs*", Math. Japonica 6, (1980) 573-577.
- [12]. B. M. Waxman, "*Routing of multipoint connections*", IEEE J. Selected Areas Commun. 6(9) (1988) 1617-1622.

# Analysis of Cell Spacing on a TCP Flow Mapped onto an ATM Connection

Fabrice Guillemin<sup>1</sup>, Jacqueline Boyer<sup>1</sup>, and Olivier Dugeon<sup>1</sup>,  
and Christophe Mangin<sup>2</sup>

<sup>1</sup> France Télécom, CNET/DAC,

2 Avenue Pierre Marzin, F-22307 Lannion Cedex

<sup>2</sup> ITE/TCL, Mitsubishi Electric, Immeuble Germanium,  
80, Avenue des Buttes de Cosmes, F-35700 Rennes

**Abstract.** We show in this paper that the cell spacing technique can be used to regulate TCP when the traffic of a TCP connection is carried over an ATM connection. For this purpose, we consider a very simple traffic configuration where two workstations exchange TCP data traffic segmented into ATM cells and transmitted over an ATM link traversing a cell spacer. We precisely analyze how the buffer content of the cell spacer evolves over time. The results obtained via very simple mathematical models are then compared with those of a real TCP data transfer experiment.

## 1 Introduction

The so-called transmission control protocol (TCP) is today certainly the transfer protocol the most implemented for data transmission in computer communications. Whereas this situation has been known for a long time in local area networks, it begins to appear for wide area networks. Indeed, with the emergence of the Web in the mid'90s and in parallel with the large scale deployment of the Internet, TCP generates today a huge volume of traffic over wide area backbone networks (see for instance [9] for statistics on Internet traffic).

To estimate the quality of service offered to a data transfer application and hence, the quality of service perceived by a customer, it is essential to model the stochastic behavior of a TCP flow through a wide area network. This task is however made extremely difficult by different factors. First, it is important to note that whatever be the transfer technology chosen, the behavior of TCP greatly depends on the specific implementation considered and in particular on the way acknowledgement segments are returned by the destination to the source. It is hence difficult to draw general conclusions, which hold for all TCP implementations, especially when the results critically depend on the traffic configuration (number of users, number of routers, transmission speed of the network links, etc.).

Different techniques may be envisaged to carry TCP/IP traffic over a wide area network, for instance directly over the transmission layer (IP over SDH or WDM) or by using a connection oriented transfer technology, such as ATM or

Frame Relay. In this paper, we focus on the transport of TCP/IP traffic over ATM. A major difficulty for taking benefit of the high quality transfer of ATM is that there is no direct link between the windowing flow control mechanism at the TCP level and the traffic control procedures at the ATM layer, which are cell oriented (e.g., leaky buckets for traffic parameter enforcement, violation tagging, congestion notification, etc.). It hence appears that analytical models for capturing the dynamics of TCP/IP over ATM in complex traffic configurations seem to be out of reach.

In spite of this difficulty, several simulation studies for the transport of TCP traffic over an ATM connection using a statistical ATM transfer capability have been carried out in the technical literature. It actually appears that the adequation of the ATM transfer capabilities to the transport of TCP critically depends on the fine tuning of control parameters (see [1] for the statistical bit rate capability, [7] for the unspecified bit rate capability with early or partial packet discard, [2] for the guaranteed frame rate capability, [5] for the available bit rate capability).

The different control mechanisms studied so far in the technical literature try to regulate TCP via packet discard. Instead, we show in this paper that a reliable method of monitoring TCP dynamics while guaranteeing no loss is to offer sufficient buffering at network access and to control TCP dynamics via queueing. Specifically, working on the principle that TCP can adapt to any constant bit rate pipe, for which transfer delays have small magnitude variations, we propose in this paper to use the cell spacing technique to regulate TCP traffic carried over an ATM connection.

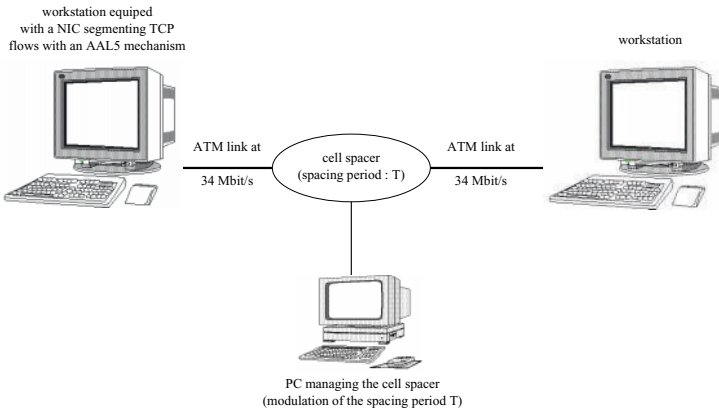
The technique explored in this paper for regulating TCP relies on ATM as transfer technology, but other techniques working directly at the TCP level have also been investigated in the literature (e.g., the Packeteer proposal [6]). Finally, let us note that further techniques consists of delaying acknowledgement segments in the destination terminal so as to limit the amount of data transmitted by the source (see [4] for instance). In that case, we could say that TCP regulation is performed in the backward direction, whereas packet shaping performs the same task on the forward direction.

The organization of this paper is as follows: In Section 2, the reference traffic configuration considered to analyze the impact of cell spacing on a TCP connection is described. The evolution of the buffer content of the cell spacer is studied in Section 3 for the slow start period and in Section 4 in the congestion avoidance regime. We then examine the case in Section 5 when acknowledgment segments are delayed, i.e., an acknowledgement is returned by the destination only when a certain number of packets have been received (namely two in the implementation analyzed). In Section 6, we compare the results obtained via the previous simple mathematical models with experimental data. Some concluding remarks are presented in Section 7. In this paper, we do not detail the description of TCP (see reference books like that by Stevens [8] for an exhaustive presentation of the TCP/IP protocol stack).

## 2 Reference Configuration and Notation

Let us consider the configuration depicted in Figure 1 and representing an idealized TCP data transfer between two workstations. A (full duplex) TCP connection is established between two identical workstations, connected to each other via an ATM link with bit rate  $\mu = 34$  Mbit/s. TCP traffic is segmented into cells in the terminals by using an AAL 5 mechanism and the resulting cell stream is transmitted over a bidirectional ATM connection.

The peak cell rate at the output of the source workstation is denoted by  $h$ . Let  $w$  be the maximum TCP window size, negotiated at the establishment of the TCP connection and expressed in segments; we assume that  $w$  is constant over time and that TCP segments encapsulated in IP packets have all the same size (corresponding to the maximum segment size), giving rise to  $\sigma$  cells.



**Fig. 1.** Experiment for the regulation of TCP via cell spacing

Let  $\delta$  denote the time for a segment issued by the source workstation to be acknowledged by the destination, that is, the corresponding acknowledgement segment is completely received by the source workstation, when the cell spacer is inactive.  $\delta$  depends on the propagation time along the ATM link, the processing time in the receiving workstation, and the bit rate of the reverse ATM connection, since an acknowledgement message gives rise, in general, to 2 cells.  $\delta$  is assumed to be a constant.

In the following, we analyse the transfer of data from one workstation (the source) to the other workstation (the destination). The cell spacer can be modeled as a single server queue fed with the cell stream generated by the source. The service rate is denoted by  $c < h$  and is equal to the spacing rate, which can be modulated over time via a PC. The time to serve a segment is  $\sigma/c$ . Hence, the round trip time  $\delta$  should be replaced with  $\Delta = \delta + \sigma/c$ .

### 3 Analysis of the Slow Start Period

Let us first analyse the slow start period; the slow start threshold is denoted by  $b$  (expressed in segments giving rise to  $b\sigma$  cells). Let  $w_0$  be the initial congestion window size (also expressed in segments and corresponding to  $w_0\sigma$  cells). We assume that  $c\Delta/(w_0\sigma) > 1$ , since  $w_0$  is in general set to a very small value (typically one segment giving rise to  $\sigma$  cells).

The evolution of the buffer content of the cell spacer over time depends on the different parameters  $\Delta$ ,  $w_0$ ,  $\sigma$ ,  $h$ ,  $c$ , and  $b$  and on the fact that the buffer of the cell spacer empties before the slow start threshold is reached or not. Despite this complexity, the outstanding property of the system is that it is possible to determine the content of the buffer of the cell spacer at the end of the slow start period, namely at time  $(n_1 + 1)\Delta$ , where

$$n_1 \stackrel{\text{def}}{=} \lceil \log_2(b/w_0) \rceil, \quad (1)$$

the slow start threshold being reached over the time interval  $[n_1\Delta, (n_1 + 1)\Delta]$ . Let moreover  $n_0$  be defined by

$$n_0 \stackrel{\text{def}}{=} \lceil \log_2(c\Delta/w_0\sigma) \rceil. \quad (2)$$

When  $n_0 \leq n_1$ ,  $[n_0\Delta, (n_0 + 1)\Delta]$  is the first time interval over which the buffer does not empty before the slow start threshold is reached. We can specifically state the following result.

**Theorem 1.** *At the end of the slow start period, the buffer content of the cell spacer oscillates between the values*

$$\mathcal{V}_b^- = (b\sigma - c\Delta)^+ \text{ and } \mathcal{V}_b^+ = (b\sigma - c\Delta)^+ + \sigma \left(1 - \frac{c}{h}\right). \quad (3)$$

The oscillating regime starts at time  $t_b$  defined by

$$t_b = (\nu + 1)\Delta + [(b - 2^\nu w_0)^+ + \mathcal{K}] \frac{\sigma}{c}, \quad (4)$$

where

$$\nu = \min\{n_0, n_1\} \quad (5)$$

$$\mathcal{K} = \left[ (b - 2^\nu w_0) \frac{2c - h}{h - c} + 1 + \frac{ch}{\sigma(h - c)} \left( \frac{2^\nu w_0 \sigma}{h} - \Delta \right)^+ \right]^+. \quad (6)$$

*Proof.* From the very definition of the slow start mechanism, a new busy period for the cell spacer buffer begins at time  $n\Delta$  as long as  $n < n_0$ , which condition means that the buffer empties over the time interval  $[n\Delta, (n + 1)\Delta]$ . From time  $n\Delta$ , an acknowledgement packet is received by the source every  $\sigma/c$  time units. For each acknowledgment packet received, the source transmits two segments during a time interval of  $2\sigma/h$ .

If  $2c \leq h$ , the buffer content increases by  $\sigma$  for each acknowledgement packet received by the source. It follows that the maximum value of the buffer content over the time interval  $[n\Delta, (n + 1)\Delta]$ ,  $n \leq \min\{n_0, n_1\}$ , is

$$2^{n-1}w_0\sigma + \sigma(1 - 2c/h), \quad (7)$$

since two segments of  $\sigma$  cells arrive at the peak rate  $h$  over each time interval  $[n\Delta + k\sigma/c, n\Delta + (k+1)\sigma/c]$  for  $k = 0, \dots, 2^{n-1}w_0 - 1$ . Thus, the buffer content increases by  $\sigma$  at the end of each time interval  $[n\Delta + k\sigma/c, n\Delta + (k+1)\sigma/c]$  for  $k = 0, \dots, 2^{n-1}w_0 - 2$  and by  $2\sigma(1 - c/h)$  at the end of the last time interval  $[n\Delta + (2^{n-1} - 1)\sigma/c, n\Delta + 2^{n-1}\sigma/c]$ .

When  $h < 2c$ , the buffer content linearly increases up to the value

$$2^n w_0 \sigma (1 - c/h). \quad (8)$$

To summarize, we can state that when  $n < \nu \stackrel{\text{def}}{=} \min\{n_0, n_1\}$ , the buffer content empties on each time interval  $[n\Delta, (n+1)\Delta]$  and the maximum buffer content is given by (8) when  $2c > h$  and (7) when  $2c \leq h$ , respectively.

Now, we examine what happens on the time interval  $[n\Delta, (n+1)\Delta]$  when  $n \geq \nu$ . The two cases  $n_0 \geq n_1$  and  $n_0 < n_1$  must be distinguished because they lead to different behaviors of the cell spacer buffer. In the following, for the sake on conciseness, we analyse only the first case ( $n_0 \geq n_1$ ); similar computations can be carried out when  $n_0 < n_1$ .

When  $n_0 \geq n_1$ , the buffer of the cell spacer may not be empty at the end of the time interval  $[n_1\Delta, (n_1+1)\Delta]$  over which the slow start threshold is reached. If  $h > 2c$ , the buffer content increases by  $\sigma$  for the first  $(b - 2^{n_1-1}w_0 - 1)$  acknowledged segments and by  $2\sigma(1 - c/h)$  for the last segment, for which the congestion window hits the slow start threshold at time

$$\tau_b = n_1\Delta + (b - 1 - 2^{n_1-1}w_0) \frac{\sigma}{c}. \quad (9)$$

Then, since an acknowledgement is received by the source every  $\sigma/c$  time units, the buffer content oscillates between the values

$$v_b^- = (b - 2^{n_1-1}w_0)\sigma \text{ and } v_b^- = (b - 2^{n_1-1}w_0)\sigma + \sigma \left(1 - \frac{c}{h}\right), \quad (10)$$

until the whole congestion window has been completely acknowledged. After this oscillating period, the buffer content goes to the value  $(b\sigma - c\Delta)^+$ .

When  $2c > h$ , the slow start threshold is still reached at time  $\tau_b$  defined by equation (9). At that time, the congestion window size is equal to  $b$  segments and  $2(b - 2^{n_1-1}w_0)$  segments can be transmitted over the time interval  $[n_1\Delta, \tau_b]$ . At time  $\tau_b + k\sigma/c$ ,  $k \geq 0$ , the source has received credits to transmit, at the peak rate  $h$ ,  $2(b - 2^{n_1-1}w_0) + k$  segments over the time interval  $(\tau_b - n_1\Delta) + k\sigma/c$ . It follows that the buffer content linearly increases as long as

$$k \frac{\sigma}{c} + (\tau_b - n_1\Delta) \leq [2(b - 2^{n_1-1}w_0) + k - 1] \frac{\sigma}{h},$$

that is,

$$k \leq k_0 \stackrel{\text{def}}{=} \left\lfloor (b - 2^{n_1-1}w_0) \frac{2c - h}{h - c} + 1 \right\rfloor$$

The buffer content ramps up to the value  $[2(b - 2^{n_1-1}w_0) + k_0]\sigma(1 - c/h)$  and then oscillates between the values  $v_b^-$  and  $v_b^+$  defined by equation (10), before going to the value  $(b\sigma - c\Delta)^+$ .

To summarize, when  $n_1 \leq n_0$ , at the end of the slow start period, the buffer content oscillates between the values

$$\mathcal{V}_b^- = (b\sigma - c\Delta)^+ \text{ and } \mathcal{V}_b^+ = (b\sigma - c\Delta)^+ + \sigma \left(1 - \frac{c}{h}\right) \quad (11)$$

in the case  $h < 2c$  as well as  $h > 2c$ . These expressions are consistent with equation (3) when  $n_1 \leq n_0$ . The oscillating regime begins at time  $t_b = (n_1 + 1)\Delta$ , which expression is consistent with equation (4). This completes the proof.

## 4 Analysis of the Congestion Avoidance Period

After the slow start period, the TCP window flow control enters the congestion avoidance regime (when, of course,  $b < w$ ). The congestion window  $cwnd$  increases by  $MSS * MSS / cwnd$  each time an acknowledgement packet is received, up to the the maximum window size  $w$ . To simplify the discussion, we assume in this paper that the congestion window increases by one segment when all the segments of the current congestion window have been acknowledged. Two segments are then sent by the source at the instant when this event occurs. This allows us to greatly simplify the mathematical computations.

Let

$$m_0 = \min\{m : (m + b)\sigma \geq c\Delta\}. \quad (12)$$

To avoid painful computations, we moreover make the following reasonable assumption:

$$(C_1) \quad \tau_b + (b - 1)\frac{\sigma}{c} > t_b$$

where  $t_b$  is defined by (4) and  $\tau_b$  is the time at which the congestion window hits the slow start threshold, given by (9) if  $n_1 \leq n_0$  and  $\tau_b = (n_0 + 1)\Delta + (b - 2^{n_0}w_0 - 1)\sigma/c$  in the case  $n_1 > n_0$ , respectively. This condition means that the buffer content begins to oscillate before the window of  $b$  segments is completely acknowledged.

**Theorem 2.** *At the end of the congestion avoidance period, the buffer content of the cell spacer oscillates between the values:*

$$\mathcal{V}_w^- = [w\sigma - c\Delta]^+ \text{ and } \mathcal{V}_w^+ = [w\sigma - c\Delta]^+ + \sigma \left(1 - \frac{c}{h}\right). \quad (13)$$

*Proof.* For the sake of clarity we consider the cases  $h < 2c$  and  $h > 2c$  separately.

*Case  $h > 2c$*  If  $h > 2c$ , the buffer content of the cell spacer increases by  $2\sigma(1 - c/h)$  when the congestion window increases by one segment.

If  $b\sigma \geq c\Delta$  (and then  $m_0 = 0$ ), the buffer of the cell spacer never empties and the buffer content oscillates between two values over some time intervals. Specifically, when the congestion window size is between  $b + m$  and  $b + m + 1$ ,



$m = 0, \dots, w - b - 1$ , the buffer content oscillates over a time interval of length  $(b + m - 1)\sigma/c$  between the values

$$b\sigma - c\Delta + m\sigma \quad \text{and} \quad b\sigma - c\Delta + m\sigma + \sigma \left(1 - \frac{c}{h}\right). \quad (14)$$

Two consecutive time intervals are separated by a transition interval of length  $\sigma/c$ , where the buffer content increases before starting an oscillation period. The buffer content eventually oscillates between the values

$$\mathcal{V}_w^- = w\sigma - c\Delta \quad \text{and} \quad \mathcal{V}_w^+ = w\sigma - c\Delta + \sigma \left(1 - \frac{c}{h}\right), \quad (15)$$

when the maximum window size is reached. This expression is consistent with equation (13).

When  $b\sigma < c\Delta$ , the buffer of the cell spacer empties over the time intervals  $[(n_1 + m)\Delta, (n_1 + m + 1)\Delta]$  for  $m = 1, \dots, m_0 - 1$ . Assume first that  $b + m_0 > w$  (i.e., the buffer always empties). On the successive time interval  $[(n_1 + k)\Delta, (n_1 + k + 1)\Delta]$ ,  $k \geq 1$ , the buffer content oscillates between 0 and  $\sigma(1 - c/h)$  before a transient period where the buffer reaches the  $2\sigma(1 - c/h)$  before returning to 0. This happens until the maximum window size is reached, and then, the buffer content eventually oscillates between the values 0 and  $\sigma(1 - c/h)$ . This result is consistent with equation (13).

In the case  $b + m_0 < w$ , the buffer of the cell spacer does not empty from time  $(n_1 + m_0)\Delta$  on, and oscillates stepwise between the values  $(b + m - 1)\sigma$  and  $(b + m - 1)\sigma + \sigma(1 - c/h)$  over the  $m$ th time interval of length  $(b + m - 1)\sigma/c$ ,  $m = m_0, \dots, w - b - 1$ . The successive oscillation intervals are separated by a transition interval of length  $\sigma/c$ , where the buffer increases before oscillating. The buffer content eventually oscillates between the values

$$\mathcal{V}_w^- = w\sigma - c\Delta \quad \text{and} \quad \mathcal{V}_w^+ = w\sigma - c\Delta + \sigma \left(1 - \frac{c}{h}\right). \quad (16)$$

This expression is consistent with equation (13).

*Case  $h < 2c$*  Now, we consider the case when  $h < 2c$  and assume in a first step that  $b\sigma < c\Delta$ . If  $b + m_0 > w$ , the buffer always empties and its behavior is identical to that described above under the same assumptions, except during some time intervals of length  $\sigma/c$  when two segments arrive at the cell spacer. If  $b + m_0 < w$ , the buffer does not empty any more from time  $(n_1 + m_0)\Delta$  on. To understand what happens when the congestion window increases by one, assume that such an event occurs at time  $t_0$ . The buffer content linearly increases at rate  $(h - c)$  over the time interval  $[t_0, t_0 + k\sigma/c]$  as long as

$$(1 + k)\frac{\sigma}{c} \leq (2 + k - 1)\frac{\sigma}{h},$$

that is,

$$k \leq \left\lfloor \frac{c}{h - c} \right\rfloor$$

The buffer content increases up to the value  $v(t_0) + (2 + k)\sigma(1 - c/h)$ , where  $v(t_0)$  is the buffer content at time  $t_0$ , and then oscillates between the values  $v(t_0) + \sigma$

and  $v(t_0) + \sigma + \sigma(1 - c/h)$ . Hence, we see that except for transition periods, the behavior of the buffer content is roughly the same as in the case  $h > 2c$ . The same conclusions hold under the assumption  $b\sigma > c\Delta$ . This completes the proof.

From the above results, we see that at the end of the congestion avoidance period, the source adapts to the spacing rate  $c$  by sending cells at a rate at most equal to  $c$ . Hence, the cell spacing technique is capable of regulating a TCP source so that it adapts its transmission rate to a prescribed spacing rate. Loosely speaking, by means of the cell spacing technique, a TCP connection can be assigned a bit rate, whereas by definition, there is no concept of bit rate in the TCP flow control mechanism, handling amounts of data only. This is a key point of TCP over ATM regulated by cell spacing.

Besides, when we consider the equations (3) and (13), we can observe the following somewhat paradoxical phenomenon. Indeed, for a fixed round trip time, when  $c$  is large, the source transmits a large amount of data, and then, we may expect that the buffer is quite full. Surprisingly, it turns out that the larger the spacing rate, less data are buffered in the cell spacer, even if the slow start mechanism is enabled. This phenomenon is due to the fact that a significant part of the congestion window is buffered in the so-called “channel memory”, which appears in the equations via the term  $c\Delta$ . In fact, two parameters have a major impact on the buffer content: the spacing rate  $c$  and the round trip time  $\Delta$ . In parallel, we can also note that, the larger the round trip time, the smaller the buffer content is.

## 5 Delayed Acknowledgements

We now assume that one acknowledgement packet is returned to the source when two segments are received by the destination. During the slow start phase, an acknowledgement packet received by the source generates three credits (two for the two acknowledged segments plus one outstanding credit due to slow start). We use the same arguments as in the previous sections to study the evolution over time of the buffer content of the cell spacer. The congestion window starts with two segments, since two segments should be received by the destination before returning an acknowledgement packet.

Assume that the first segment arrives at the buffer of the cell spacer at time 0. Then, on the time interval  $[0, \Delta']$ , where  $\Delta' = \delta + 2\sigma/c$ , two segments arrive at the cell spacer. In the time intervals  $[\Delta', 2\Delta']$  and  $[2\Delta', 2\Delta' + \Delta]$ , where  $\Delta = \delta + \sigma/c$ , three segments arrive. On the time interval  $[2\Delta' + \Delta, 3\Delta' + \Delta]$ , two bursts of three segments arrive, the distance between the beginning of the two consecutive bursts is equal to  $2\sigma/c$ . On the time interval  $[3\Delta' + \Delta, 4\Delta' + \Delta]$ , nine segments arrive, and so on.

In general, as long as the buffer of the cell spacer empties, we can note that a certain number of segments arrive in bursts of three segments over a time interval with a certain length, equal to either  $\Delta'$  or  $\Delta$ . To be more specific, let  $p_n$ ,  $L_n$ , and  $c_n$  denote the number of segments and the length of the  $n$ th time interval, and the number of segments of the  $(n - 1)$ th time interval, which have

not been acknowledged, respectively. It is clear that  $c_n = 0$  if all the segments have been acknowledged and  $c_n = 1$  if one segment has not been acknowledged. If  $c_n = 0$ , then  $L_n = \Delta'$  and if  $c_n = 1$ , then  $L_n = \Delta$ . It can moreover be shown that as long as the buffer of the cell spacer empties,  $p_n$  and  $c_n$  and  $L_n$  satisfy the following recursion:

$$n \geq 1, p_{n+1} = 3c_n + 3 \left\lfloor \frac{p_n - c_n}{2} \right\rfloor \text{ and } c_{n+1} = p_n - c_n - 2 \left\lfloor \frac{p_n - c_n}{2} \right\rfloor \quad (17)$$

with  $p_1 = 2$  and  $c_1 = 0$ .

Over the  $n$ th time interval for  $n \geq 2$ ,  $p_n/3$  bursts of three segments at the peak cell rate  $h$  arrive at the cell spacer. Each segment is composed of  $\sigma$  cells and the distance between the beginning of two consecutive bursts is equal to  $2\sigma/c$ . It follows that the maximum buffer size over the  $n$ th time interval is equal to

$$v_n = \left( \frac{p_n}{3} - 1 \right) \sigma + 3\sigma \left( 1 - \frac{c}{h} \right) \quad (18)$$

if  $3c < 2h$ , and

$$v_n = p_n \sigma \left( 1 - \frac{c}{h} \right) \quad (19)$$

if  $3c > 2h$ . The server of the buffer of the cell spacer is busy over a time interval of length  $p_n \sigma / c$ . As long as  $p_n \sigma \leq cL_n$ , the buffer of the cell spacer empties over the  $n$ th time interval.

As in the previous sections, let us introduce the integers  $n_0$  and  $n_1$  defined by

$$n_0 = \min\{n : p_n \sigma > cL_n\}, \quad (20)$$

$$n_1 = \min\{n : p_n > b\}. \quad (21)$$

The  $n_0$ th time interval is the first interval over which the buffer of the cell spacer does not empty and the  $n_1$ th time interval is the first interval over which the slow start threshold is reached. Let  $m_0$  be defined as in equation (12). The analysis of the previous section can be extended and we can state the following result, whose proof is omitted.

**Theorem 3.** *At the end of the slow start period, the buffer content oscillates between the values*

$$\mathcal{V}_b^- = (b\sigma - cL_{n_0})^+ \text{ and } \mathcal{V}_b^+ = (b\sigma - cL_{n_0})^+ + 2\sigma \left( 1 - \frac{c}{h} \right). \quad (22)$$

*Once the maximum congestion window size  $w$  has been reached, the buffer content oscillates between the values*

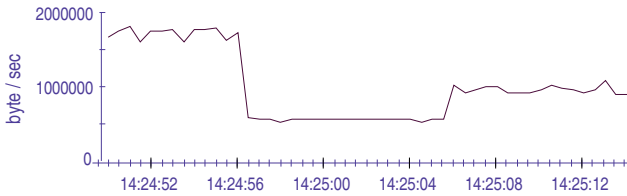
$$\mathcal{V}_w^- = [w\sigma - cL_{n_1+m_0}]^+, \mathcal{V}_w^+ = [w\sigma - cL_{n_1+m_0}]^+ + 2\sigma \left( 1 - \frac{c}{h} \right). \quad (23)$$

The above formulae will be compared in the next section with what is observed via an experimentation with a real TCP/IP protocol stack.

## 6 Experimental Results

To show the capability of cell spacing to regulate TCP dynamics without information loss in a real situation, the configuration depicted in Figure 1 has been used with two Silicon Graphics workstations connected to each other through a cell spacer via a 34 Mbit/s link. The operating system on the workstations is Irix 6.3, the TCP protocol stack is BSD 4.4, the slow start mechanism is enabled, and the TCP retransmission time out value is initially set to 1 second. The TCP information flow generated by a workstations is segmented in a NIC (network interface card) implementing an AAL 5 mechanism. The maximum segment size MSS is equal to 1516 bytes and the maximum window size is equal to  $60 \times 1024$  bytes, corresponding to 41 segments with maximum size.

The cell spacer is managed via a PC, which allows the cell spacing period to be modulated over time. A buffer space of 1300 cells is dedicated to the connection in the cell spacer. The dynamics of TCP are observed by using the Silicon Graphics Performance Co-Pilot (v. 2.0) tool. The TCP application consists in displaying from the server workstation a graphical animation sequence on the receiving workstation through an Xwindow client server connection.



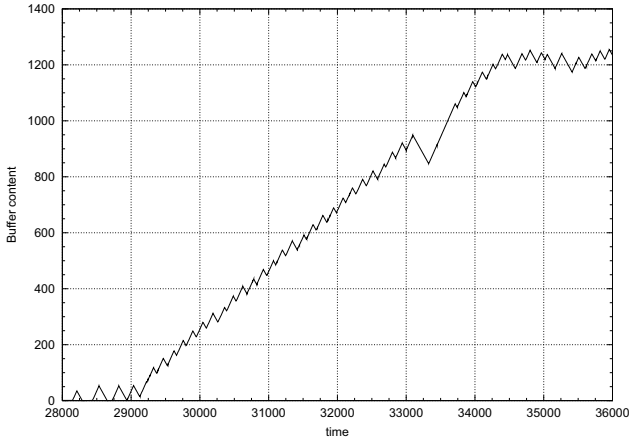
**Fig. 2.** Instantaneous bit rate of the TCP application when the cell spacing period is modulated

Figure 2 shows the adaptation of TCP to changes of the spacing period, that is, how the bit rate of the TCP connection fits the spacing rate. At the beginning the spacing rate is set equal to 33.920 Mbit/s and the TCP application is not constrained (its free run rate is then about 2 Mbyte/s). The spacing rate is then set equal to 5 Mbit/s and TCP adapts its transmission rate to 0.5 Mbytes/s, without information loss. In the last part of Figure 2, the spacing rate is set equal to 10 Mbit/s and TCP regulates at 1 Mbyte/s. This clearly shows that TCP can be regulated via cell spacing .

With regard to the quality of service, when the cell spacing rate is decreased, the perceived quality of the graphical animation sequence is of course not as good as in the case of free run, but is still satisfying. In particular, there is no interruptions due to packet loss. The movement of the objects is just slowed down.

To conclude, let us examine the behavior of the buffer content of the cell spacer. In the TCP/IP protocol stack considered, one acknowledgement packet

is returned to the source when two segments are received. The evolution of the buffer content (expressed in cells) is depicted in Figure 3, when the spacing rate is about  $c = 15.264$  Mbit/s. The time unit is the cell transmission time, namely 12,5 microseconds. The round trip time is  $\Delta = 3.6$  milliseconds. A TCP segment gives rise to  $\sigma = 32$  cells. The slow start threshold is equal to the maximum window size, equal to  $w = 1312$  cells.



**Fig. 3.** Content of the buffer of the cell spacer plotted from experimental data.

In Figure 3, we can distinguish the first time intervals where the buffer of the cell spacer empties. Then, when the buffer does not empty anymore, the buffer content increases almost linearly at rate  $c/2$  (because  $3\sigma$  cells are transmitted every  $2\sigma/c$  and the buffer release rate is equal to  $c$ ) until it hits the maximum window size and then oscillate between some values, which are not very far from the theoretical values

$$w - c\Delta = 118 \quad \text{and} \quad w - c\Delta + 2\sigma(1 - c/h) = 1217. \quad (24)$$

The difference between the theoretical and the actual values is due to the real behavior of TCP and to round off errors.

## 7 Conclusion

In this paper, we have shown that the cell spacing technique can be used to efficiently regulate a TCP connection without packet loss, provided that the cell spacer can buffer at least a whole TCP window per connection. In fact, cell spacing acts in such a way that TCP automatically adapts to the offered constant bit rate pipe, which bit rate is equal to the spacing rate.

Such a regulation technique can then be combined to a bandwidth reservation and renegotiation scheme to transport TCP traffic over variable bit rate connections. Among all the resource reservation techniques, which have been specified for ATM networks, the resource management (RM) procedures, which operate on the same time scale as the round trip time across the network, offer the best response time with minimal overhead. Their responsiveness allows in particular a coupling between the establishment of a TCP connection with that of the underlying ATM connection [3].

Such an approach is very close to that followed by the MPLS group within the IETF, but it includes in addition traffic management aspects, which are currently not covered by MPLS, such as the regulation of TCP via cell spacing and the explicit reservation of resources (namely bandwidth in the network and buffer space in the cell spacer) for carrying TCP traffic. In this respect, it may be envisioned to use the label distribution protocol to set up underlying ATM connections. This issue will be addressed in future work.

## References

1. Bonaventure O., Klovning E., and Danthine A (1996) Is VBR a solution for an ATM LAN ? In *Proc. IFIP 5th International Workshop on Protocols for High Speed Networks*, pp. 60–74.
2. Bonaventure O. (1997) A simulation study of TCP with the proposed GFR service category. In *Proc. High Performance Networks for Multimedia Applications*.
3. Boyer J., Boyer P. , Dugeon O., Guillemin F., and Mangin C. (1999) Accelerated signalling for the Internet over ATM. *European Transactions on Telecommunications and Related Technologies* 10(2): 153-164.
4. Ghani N. (1999) Enhanced TCP/IP ACK pacing for high speed networks. In *Proc. ITC'16*.
5. Johansson P., Wedlund E., and Karlsson J. (1997). Interaction between TCP flow control and ABR rate control. In *Proc. IEEE ATM'97 Workshop*.
6. The Packeteer Technical Forum (1998) TCP/IP bandwidth management series.
7. Romanow A. and Floyd S. (1994) Dynamics of TCP traffic over ATM networks. In *Proc. ACM Sigcom'94*, pp. 79–88.
8. Stevens W.R. (1994) *TCP/IP illustrated*. Vol. 1. Addison Wesley Professional Computing Series.
9. Thompson K., Miller G.J., and Wilder R. (1997) Wide area Internet traffic patterns and characteristics. *IEEE Network Magazine*, November/December issue.

# Mapping of Loss and Delay Between IP and ATM Using Network Calculus

Tijani Chahed, Gérard Hébuterne, and Caroline Fayet

Institut National des Télécommunications  
Telecommunication Networks and Services Dept.  
9 rue C. Fourier, 91011 Evry CEDEX, France  
{tijani.chahed, gerard.hebuterne, caroline.fayet}@int-evry.fr

**Abstract.** End-to-end QoS requires accurate mapping of QoS classes and particularly QoS parameters between the different, heterogeneous layers and protocols present at the terminal equipment and intermediate nodes. In the IP over ATM context, both technologies offer QoS capabilities but differ in architecture, service classes and performance parameters. In this paper, we consider mapping of loss and delay between Integrated Services IP and ATM, using Network Calculus (NC), a min-plus algebra formulation. NC targets lossless systems only and is thus suitable for guaranteed services. However, as our aim is to quantify and map loss and delay in the presence of loss, we extend the theory so as to account for lossy systems too, as shall be the case for non-conformant traffic and congested systems. Loss is introduced by setting constraints on both the delay and the buffer size at the network elements. Numerical applications are used to further illustrate the mapping of loss and delay between IP and ATM.

## 1 Introduction

QoS is composed of two closely related components: a user-oriented (objective and subjective) set of QoS parameters reflecting a user QoS requirement and network-oriented (objective) bounds on the QoS parameters to satisfy the requirement [1]. A correct and accurate (qualitative and quantitative) relationship between the two components is central to the overall, end-to-end QoS performance.

The approach adopted so far [2] is to map the user-oriented QoS parameters directly into a Network Performance (NP), measured at the network layer in terms of performance parameters of significance to the network provider only. The transfer parameters have been defined for the network layer only, be it for ATM [3] or IP [4,5]. This undermines the various heterogeneous layers superposed at the terminal equipment as well as in intermediate nodes and the protocol stack found therein [7]. As noted in [6], QoS and its corresponding bounds on the QoS parameters should be defined for each layer and translated to the next layers.

Both IP and ATM aim to offer QoS but differ in architecture and mechanisms. QoS in ATM is based on classes which are themselves based on the

services they offer (e.g., real-time) and traffic profiles (e.g., CBR, VBR). For IP, two QoS architectures have been defined: Integrated Services (intserv) [9] and Differentiated Services (diffserv) [10].

Intserv is based on RSVP signaling and offers a means to a per-flow QoS in IP. Three service categories have been defined: Guaranteed (GS), Controlled-Load (CLS) and Best Effort (BE). Owing to the presence of ATM at the lower layers and at the core of networks, a framework for translation of QoS between intserv and ATM is set in [11] and the interoperation of intserv classes with ATM is investigated in [12]. However these translations are valuable for mapping IP service classes to corresponding ATM QoS classes, they do not offer quantitative nor qualitative correspondence between each QoS parameter in IP and ATM. The performance parameters and their mapping at each layer provide firm basis to map the two approaches and hence guarantee end-to-end QoS [8].

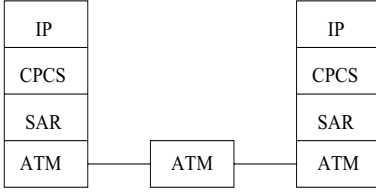
The aim of this work is precisely to map QoS parameters, namely loss and delay, between the IP and ATM layers. As far as deterministic bounds are concerned, we choose to use Network Calculus (NC) [13], based on min-plus algebra, to reach this aim. NC has been defined for lossless queuing systems only. As such, it is useful to study lossless, guaranteed services. However, as our aim is to quantify loss and delay in the presence of loss, we extend NC theory so as to account for lossy systems too, as shall be the case for non-conformant GS and CLS traffic and BE traffic in the presence of congestion. Let us note that in an independently different context, work on the representation of the loss process using NC has been formulated in [14], in a fashion that should not interfere with the present work.

The remainder of this paper is organized as follows. In Section 2, we set the end-to-end system where IP and ATM modules co-exist. This case may arise either at the terminal equipment or some intermediate node between IP and ATM subnetworks. In this context, we study the NC formulation for maximum delay and backlog for an intserv - GS service, its transition to an ATM system and quantify the cost of a cascade of protocol stacks as opposed to a concatenation of the two systems. In Section 3, we introduce loss in NC and specifically define the regions of loss and the bounds the formulation offers. Next, we obtain representations for loss for both the constrained-buffer and the constrained-delay cases. In Section 4, we present numerical results for mapping of loss and delay between the IP and ATM layers. Conclusion and perspectives of future work are given in Section 5.

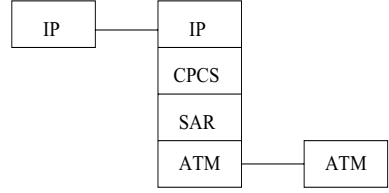
## 2 End-to-End System

Our focus is on systems that integrate both IP and ATM modules. This case arises either at the end equipment (Figure 1) or intermediate nodes between IP and ATM subnetworks (Figure 2).





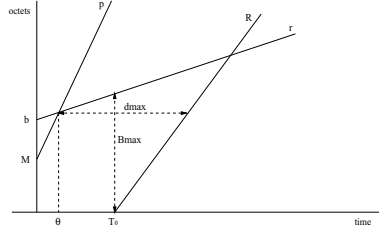
**Fig. 1.** End-to-end System: IP over ATM in terminal equipment



**Fig. 2.** End-to-end System: IP over ATM in intermediate node

## 2.1 Intserv - GS

We investigate the intserv - GS class. The arrival curve has the following Tspec (see Figure 3) as given by a token bucket and peak rate model:  $M$  (maximum packet size),  $p$  (peak rate),  $r$  (sustainable rate) and  $b$  (burst tolerance).



**Fig. 3.** Intserv - GS

Mathematically, the arrival curve is:  $\alpha(t) = \min(pt + M, rt + b)$ , the service curve is:  $c(t) = R(t - T_0)^+$ , and the output curve is:  $\alpha^*(t) = \min(R(t - T_0), rt + b)$  for  $t > T_0$ . We now derive expressions for maximum delay  $d_{max}$ , as implied by NC. First, we determine  $\theta$ . We have  $r\theta + b = p\theta + M$ , so,  $\theta = \frac{b-M}{p-r}$ . To determine  $d_{max}$ , we distinguish three cases:

$$d_{max} = \begin{cases} T_0 + \frac{M}{r} & \text{at } t = 0 \text{ if } R > p \\ \frac{p-R}{R}\theta + \frac{M}{R} + T_0 & \text{at } t = \theta \text{ if } p > R > r \\ \infty & \text{if } R < r \end{cases}$$

We now determine the maximum backlog  $B_{max}$ . We distinguish two cases :

1. if  $T_0 > \theta$  ;

$$B_{max} = \begin{cases} rT_0 + b & \text{at } t = T_0 \text{ if } R > r \\ \infty & \text{if } R < r \end{cases}$$

2. if  $T_0 < \theta$  ;

$$B_{max} = \begin{cases} pT_0 + M & \text{at } t = T_0 \text{ if } R > p \\ (p - R)\theta + M + RT_0 & \text{at } t = \theta \text{ if } p > R > r \\ \infty & \text{if } R < r \end{cases}$$

## 2.2 Transition

The transition from one system to the next, needs to translate the Tspec of intserv - GS to appropriate, corresponding ATM rtVBR's pair of leaky buckets formulation, taking into account the overhead ratio and its implications between the layers, as shown in Figure 4 when the two systems are in cascade.

The overhead ratio  $O_i$  is the ratio of the (expected) frame length  $L_i$  at layer  $i$  to the (expected) frame length  $L_{i-1}$  at the previous, adjacent layer  $i-1$  (following the direction of flow), and indicates the fractional amount of traffic that is being added (subtracted) as traffic is flowing down (up) the layers at the transmitter (receiver). Formally, at the receiver,  $O_i = \frac{L_i}{n_{i-1} \times L_{i-1}}$ , where  $n_i$  is the number of segments resulting from segmentation of the frame at layer  $i$  [8].  $O_i$  is larger than 1 at the sender. An inverse relation governs the receiver, where  $O_i$  is smaller than 1.

For a system  $i$  with Tspec  $(M_i, p_i, r_i, b_i)$  and service components  $(R_i, T_{0,i})$ , the following transition rules hold for  $i > 0$ : i.  $M_{i+1} = O_i$ , ii.  $b_{i+1} = b_i \times O_i$ , iii.  $r_{i+1} = r_i \times O_i$  and iv.  $p_{i+1} = R_i$ . The service curve itself needs to account for the overhead ratio as the capacity at each layer needs to be accurately translated from the link layer capacity. Hence,  $R_{i+1} = R_i \times O_i$ .

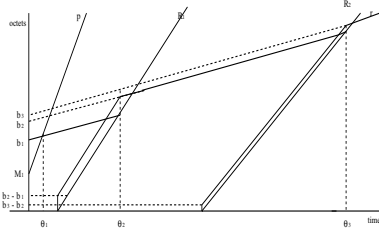


Fig. 4. Cascade

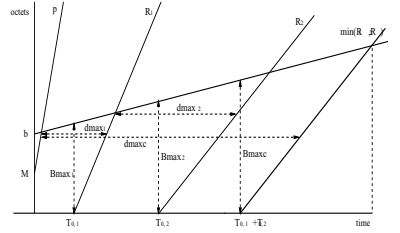


Fig. 5. Concatenation

## 2.3 Next System

Based on the NC formulation for system  $i$  and the rules for transition, we develop the formulae for system  $i+1$ .

For a traffic descriptor :  $R_{i+1}$  (peak rate),  $r_{i+1}$  (sustainable rate),  $M_{i+1}$  and  $b_{i+1}$ , where  $R_{i+1} = R_i O_i$ ,  $r_{i+1} = r_i$ ,  $M_{i+1} = -R_i T_{0,i} O_i$  and  $b_{i+1} = b_i O_i$ , the input curve is:  $\alpha_{i+1}(t) = \min(p_{i+1}t + M_{i+1}, r_{i+1}t + b_{i+1})$  for  $t > T_{0,i}$ , the service curve is:  $c_{i+1}(t) = R_{i+1}(t - T_{0,i+1})^+$ , and the output curve :  $\alpha_{i+1}^*(t) = \min(p_{i+2}t + M_{i+2}, r_{i+2}t + b_{i+2})$  for  $t > T_{0,i+1}$ , where the above-mentioned transitions hold. For  $\theta_{i+1} = \frac{b_{i+1} - M_{i+1}}{p_{i+1} - r_{i+1}}$ , the same quantities derived in Section 2.1. for  $d_{max}$  and  $B_{max}$  are recovered.

## 2.4 Concatenation

To have an indication on the cost of the superposition of the IP over ATM protocol stack, the cascade of the several layers may be replaced by a concatenation that acts as a single network element. One possible concatenation scenario is a network element with a service curve  $c(t) = R_{\min}(t - T_{0,\text{sum}})^+$  where  $R_{\min} = \min(R_1, R_2, \dots, R_n)$  and  $T_{0,\text{sum}} = \sum_i^n T_{0,i}$ , as shown in Figure 5.

For an arrival curve  $\alpha(t) = \min(pt + M, rt + b)$ , the output curve is:  $\alpha^*(t) = \min(R_{\min}t + M_n, rt + b_n)$  for  $t > T_{0,n}$  where  $M_n = M \prod_i^{n-1} O_i$  and  $b_n = b \prod_i^{n-1} O_i$ .  $\theta_{\text{concat}}$  is again equal to  $\frac{b-M}{p-r}$ . For  $p > R_n > r$ ,  $d_{\text{max}} = \frac{p-R_{\min}}{R_{\min}}\theta_{\text{concat}} + \frac{M_n}{R_{\min}} + T_{0,\text{sum}}$  at  $t = \theta_{\text{concat}}$ , and  $B_{\text{max}} =$

$$\begin{cases} b + rT_{0,\text{sum}} & \text{at } t = T_{0,\text{sum}} \text{ if } T_{0,\text{sum}} > \theta_{\text{concat}} \\ p\theta_{\text{concat}} + M_n - R_{\min}(\theta_{\text{concat}} - T_{0,\text{sum}}) & \text{at } t = \theta_{\text{concat}} \text{ if } T_{0,\text{sum}} < \theta_{\text{concat}} \end{cases}$$

## 3 Loss

We now introduce loss in NC. We consider the case of a greedy source, i.e., a source with an input function equal to the arrival curve. In addition to its relative simplicity, this case offers interesting insights into worst-case analysis.

### 3.1 Intervals of Loss

Consider one system. Let us recall that the arrival curve is:  $\alpha(t) = \min(pt + M, rt + b)$  for  $t > 0$  and the service curve is:  $c(t) = R(t - T_0)^+$ . In case of no loss, the output curve is:  $\alpha^*(t) = \min(R(t - T_0), rt + b)$  for  $t > T_0$ .

Let  $Bh(t)$  denote the backlog at time  $t$  in a hypothetical infinite capacity queue with the same arrival and service curves as above.  $Bh(t) = \min(0, \alpha(t) - c(t))$ . Loss takes place when  $Bh(t)$  exceeds the actual, *finite* buffer capacity.

The interval of loss corresponds to all  $t$  where loss occurs, and an upper bound on the number of lost packets at time  $t$  in the interval of loss is the difference between  $Bh(t)$  and the actual buffer size, as suggested by [18]. A detailed derivation is found in Sections 3.3 and 3.4.

### 3.2 Bounds on Loss

Theorem 2 of [15] states that, under appropriate assumptions, for a flow with input function  $R(t)$ , constrained by an arrival curve  $\alpha$ , a service curve  $\beta$  and an output function  $R^*(t)$ , the backlog  $R(t) - R^*(t)$  satisfies for all  $t$ :  $R(t) - R^*(t) \leq \sup_{s \geq 0} (\alpha(s) - \beta(s))$ . A greedy source is one for which  $R(t) = \alpha(t)$ . Let  $Bh_{\text{max}}$  be the maximum backlog of a hypothetical infinite capacity queue, i.e.  $Bh_{\text{max}} = \sup_{s \geq 0} (\alpha(s) - \beta(s))$ .

Now, in the loss interval, for an input function  $R_l(t)$ , constrained by an arrival curve  $\alpha_l$ , a service curve  $\beta$  and an output function  $R_l^*(t)$ , the backlog  $Bl(t) = R_l(t) - R_l^*(t)$  satisfies for all  $t$ :  $R_l(t) - R_l^*(t) \leq \sup_{s \geq 0} (\alpha_l(s) - \beta(s))$ .

In this case,  $Bl_{max} = \sup_{s \geq 0} (\alpha_l(s) - \beta(s))$ . At a time of loss  $t$ , an upper bound on loss is  $Bh(t) - Bl(t)$ . An upper bound on the lost volume at a single point of time of all the interval of loss is  $Bh_{max} - Bl_{max}$ .

Moreover, we introduce a probability-of-loss quotient with an upper-bound equal to  $\frac{Bh_{max} - Bl_{max}}{Bh_{max}}$ . If the loss pattern is not cyclic, loss happens only once and the probability of loss tends to zero thereafter. However, this term obeys to the same cyclic nature as the  $B_{max}$  and  $d_{max}$  terms of NC.

There are two approaches to loss. Loss may either be caused by constraints on the buffer size or due to constraints on the delay.

### 3.3 Constrained Buffer Size

**Loss** Let  $B_l$  ( $0 < B_l < B_{max}$ ) be the buffer size at the ATM switch. Schematically, the idea is to move the line  $y = B_l$  along the y-axis.

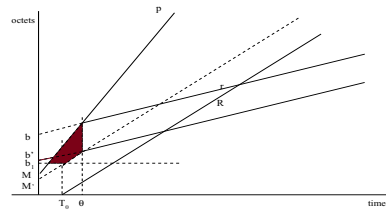
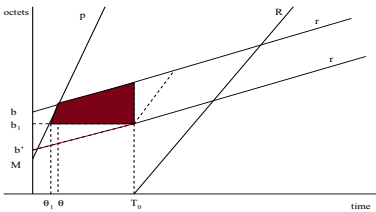
Let  $(\theta_l, b_l)$  be the point in time where loss occurs for the first time.

$$(\theta_l, b_l) = \begin{cases} (\frac{B_l - b}{p}, B_l) & \text{if } B_l > b \\ (\frac{B_l - M}{p}, B_l) & \text{if } M < B_l < b \\ (0, B_l) & \text{if } B_l < M \end{cases}$$

Let  $(b_e, \theta_e)$  be the point in time where loss occurs for the last time. We distinguish two cases :

1. if  $\theta < T_0$  (see Figure 6),  $\theta_e = T_0$  and  $b_e = B_l$  and

Instantaneous Loss (t) =  $\frac{\min(pt+M, rt+b) - B_l}{\min(pt+M, rt+b)}$  for  $\theta_l < t < T_0$  and 0 otherwise.



**Fig. 6.** Buffer-constrained Loss -  $T_0 > \theta$  **Fig. 7.** Buffer-constrained Loss -  $T_0 < \theta$

2. if  $\theta > T_0$  (see Figure 7),  $\theta_e = \theta$  and  $b_e = r\theta + b'$  where  $b' = b - (r\theta_e + b - R(\theta_e - T_0) - B_l)$ .

$$\text{Instantaneous Loss (t)} = \begin{cases} \frac{pt+M-B_l}{pt+M} & \text{for } t \in [\theta_l, T_0) \\ \frac{pt+M-R(t-T_0)-B_l}{pt+M} & \text{for } t \in [T_0, \theta] \end{cases}$$

In general,

$$\text{Inst. Loss (t)} = \frac{\min(pt+M, rt+b) - \max(B_l, R(t-T_0) + B_l)}{\min(pt+M, rt+b)} \quad (1)$$

for  $\theta_l < t < \max(T_0, \theta)$ .

**System Parameters** The arrival curve is :  $\alpha(t) = \min(pt + M, rt + b)$  for  $t > 0$ . The output curve is :  $\alpha(t)^* = \min(rt + b', R(t - T_0))$  for  $t > T_0$ , where  $b' = b - (r\theta_e + b - R(\theta_e - T_0) - B_l)$ .

**Maximum Backlog**  $B_{max} = B_l$ .

**Maximum Delay**

$$d_{max} = \begin{cases} \frac{p-R}{R}\theta + \frac{M}{R} + T_0 & \text{at } t = \theta \text{ if } B_l > b \\ \frac{p-R}{R}\theta_l + \frac{M}{R} + T_0 & \text{at } t = \theta_l \text{ if } B_l < b \end{cases}$$

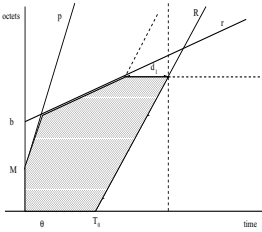
### 3.4 Constrained Delay

There are two approaches to loss caused by constrained-delay. Traffic exceeding the delay constraint is discarded (lost) after service or before entering the buffer.

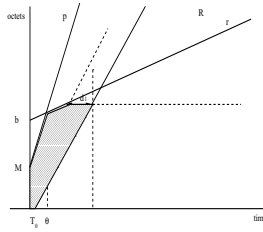
**Loss** Let  $d_l$  ( $0 < d_l < d_{max}$ ) be the maximum delay tolerated at the ATM switch. Schematically, the idea is to vary the line  $R(t + d_l - T_0)$  along the time axis.

Let  $(\theta_l, b_l)$  be the point in time where loss occurs for the first time. We distinguish three cases (see Figures 8, 9 and 10):

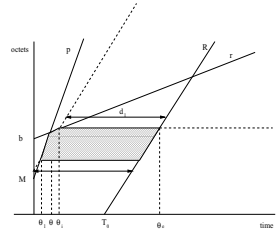
$$(\theta_l, b_l) = \begin{cases} (T_0 - d_l, R(\theta_l + d_l - T_0)) & \text{if } d_l < T_0 \\ (\max(0, T_0 - d_l), R(\theta_l + d_l - T_0)) & \text{if } T_0 < d_l < \frac{M}{R} + T_0 \\ (\frac{R(d_l - T_0) - M}{p - R}, R(\theta_l + d_l - T_0)) & \text{if } d_l > \frac{M}{R} + T_0 \end{cases}$$



**Fig. 8.** Delay-constrained Loss -  $d_l < T_0$



**Fig. 9.** Buffer-constrained Loss -  $T_0 < d_l < \frac{M}{R} + T_0$



**Fig. 10.** Buffer-constrained Loss -  $d_l > \frac{M}{R} + T_0$

Let  $(\theta_i, b_i)$  be the point in time where first octet is not lost for the first time. It is given as the intersection of  $y = R(t + d_l - T_0)$  and  $y = rt + b$ .

So,  $\theta_i = \frac{b - R(d_l - T_0)}{R - r}$  and  $b_i = R(\theta_i + d_l - T_0)$

Let  $(\theta_e, b_e)$  be the point in time where loss occurs for the last time. It is given as the intersection of  $y = R(t - T_0)$  and  $y = b_i$ .

So,  $\theta_e = \frac{d_l}{R} + T_0$  and  $b_e = R(\theta_e - T_0)$

Now, for  $d_l < \frac{M}{R} + T_0$ ,

$$\text{Inst. Loss (t)} = \begin{cases} 1 & \text{if } t \in [\theta_l, \theta_i] \\ \frac{b_e - \max(0, R(t - T_0))}{rt + b - \max(0, R(t - T_0))} & \text{if } t \in [\theta_i, \theta_e] \end{cases} \quad (2)$$

For  $d_l > \frac{M}{R} + T_0$ ,

$$\text{Inst. Loss (t)} = \begin{cases} \frac{\min(pt + M, rt + b) - b_l}{\min(pt + M, rt + b) - \max(0, R(t - T_0))} & \text{if } t \in [\theta_l, \theta_i] \\ \frac{b_e - b_l}{rt + b - \max(0, R(t - T_0))} & \text{if } t \in [\theta_i, \theta_e] \end{cases} \quad (3)$$

**System Parameters** In the case of delay-constrained loss, packets (or cells) that exceed the delay, may either be discarded at the entrance of the buffer or after service. We consider both cases :

1. Loss After Service

The arrival curve  $\alpha(t)$  is thus:  $\alpha(t) = \min(pt + M, rt + b)$ . The output curve is:  $\alpha^*(t) = \min(rt + b, R(t - T_0))$  for  $t > T_0$ .

The output curve (corresponding to non lost traffic) is:

$$\alpha^*(t) = \begin{cases} R(t - T_0) & \text{for } T_0 < t < \theta_l + d_l \\ \min(rt + b, R(t - T_0) - b_l) & \text{for } t > \theta_e \end{cases}$$

2. Loss Before Service

The arrival curve  $\alpha(t)$  in this case is:

$$\alpha(t) = \begin{cases} pt + M & \text{for } 0 < t < \theta_l \\ rt + b & \text{for } t > \theta_i \end{cases}$$

The output curve (corresponding to non lost traffic as traffic is lost before service) :

$$\alpha^*(t) = \begin{cases} R(t - T_0) & \text{for } T_0 < t < \theta_l + d_l \\ \min(rt + b, R(t - T_0) - b_l) & \text{for } t > \theta_e \end{cases}$$

### Maximum Backlog

$$B_{max} = \begin{cases} rt + b - R(t - T_0) & \text{at } t = \theta_e \text{ if } d_l < \frac{M}{R} + T_0 \\ b_l & \text{at } t = T_0 \text{ if } T_0 > \theta \text{ and } d_l > \frac{M}{R} + T_0 \\ b_l - R(\theta - T_0) & \text{at } t = \theta \text{ if } T_0 < \theta \text{ and } d_l > \frac{M}{R} + T_0 \end{cases}$$

**Maximum Delay**  $d_{max} = d_l$ .

### 3.5 Note on Packet Discard Mechanisms

Loss of some ATM cells leads to loss of IP packets, depending on the discard mechanism, be it Partial Packet Discard (PPD), Early Packet Discard (EPD) [16] or ATM-version of (Random Early Discard) RED [17]. In the case of PPD, for instance, loss of at least one ATM cell results in loss of the whole IP packet to which they belong. PPD is used in our numerical simulations, considered next.

## 4 Numerical Application

### 4.1 System Parameters

We consider the end-to-end system as shown in Figure 1. Let the source have the following characteristics:  $M = 9180$  octet;  $b = 18360$  octet;  $p = 180$  Mbps;  $r = 100$  Mbps. Let  $C_0 = 155$  Mbps and  $T_0 = 0$  with a propagation delay before and after the ATM switch corresponding to 0.01 time units.

For each layer, Table 1 summarizes the length of the SDU found in each layer as well as the overhead the layer introduces, in terms of header, trailer and/or padding.  $R_i$  is the link capacity available to each flow, on the basis of 155 Mbps and taking into consideration the overhead ratio, at each layer  $i$ .

Let us note that in the following results,  $B_{max}$ ,  $d_{max}$  and the loss ratios are given in octet, normalized to the length of frames at each layer.

### 4.2 Buffer-Constrained Performance

Let  $B_l = 1000000/53 \approx 18861$  cells at the central ATM switch.

Figure 11 shows the maximum backlog for both the non-constrained case, i.e., no loss, (curve 'B') and the buffer-constrained case (curve 'BB'). The maximum backlog at each network element or layer, as well as its mapping, indicates the dimensioning of the buffer size to be provisioned at each subsystem in order to guarantee a no loss or a constrained loss performance.

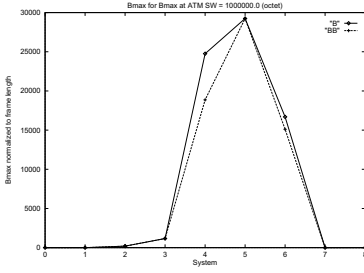
Figure 12 shows the maximum delay at each system component for both the non-constrained case, i.e., no loss, (curve 'd') and the buffer-constrained case (curve 'dB'). This yields the mapping of delay between the different layers of the model.

Figure 13 shows the loss ratio resulting from the buffer constraint at the central ATM switch (System 4) and its mapping to the next layers.

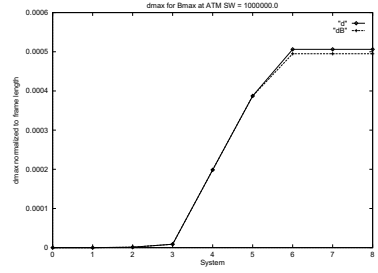
The concatenation of all the components is as follows. In case of no loss,  $R = 139.83$  and  $T_0 = 0.09$ ;  $d_{max} = 0.09$  and  $B_{max} = 9183600.0$ . The cumulative

**Table 1.** Overhead Ratio and Translation of Capacity Between Layers

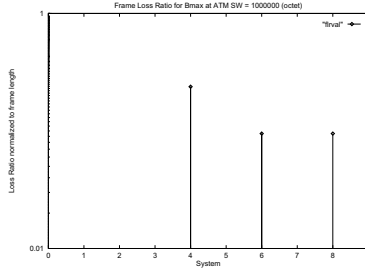
System	Layer $i$	Length $i$	$O_i$	$R_i$
0	IP	9180	1.020	171.817
1	CPCS	9216	1.004	171.146
2	SAR	48	1.000	171.146
3	ATM	53	1.104	155.000
4	ATM	53	1.000	155.000
5	ATM	53	1.000	155.000
6	SAR	48	0.906	140.377
7	CPCS	9216	1.000	140.377
8	IP	9180	0.996	139.829



**Fig. 11.** Unconstrained vs. buffer-constrained maximum backlog



**Fig. 12.** Unconstrained vs buffer-constrained maximum delay



**Fig. 13.** Buffer-constrained loss ratio

values are  $d_{max} = 0.12$  and  $B_{max} = 12955812.59$ . In case of loss,  $d_{max} = 0.09$  and  $B_{max} = 9183600.0$ . The cumulative values are  $d_{max} = 0.12$  and  $B_{max} = 12939323.4$ .

### 4.3 Delay-Constrained Performance

Let  $d_l = 0.005/53 \approx 9.43 \times 10^{-5}$  at the central ATM switch.

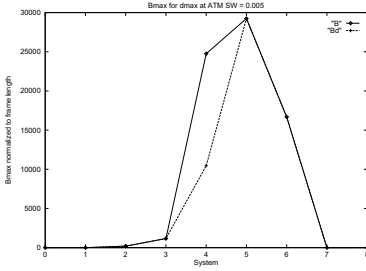
Figure 14 shows the maximum backlog for both the non-constrained case (curve 'B') and the delay-constrained case (curve 'Bd'). This again indicates the buffer needed at each subsystem to have no loss or to limit loss to a given value.

Figure 15 shows the maximum delay at each system component for both the non-constrained case (curve 'd') and the delay-constrained case (curve 'dδ'), yielding the mapping of delay between the different layers of the model.

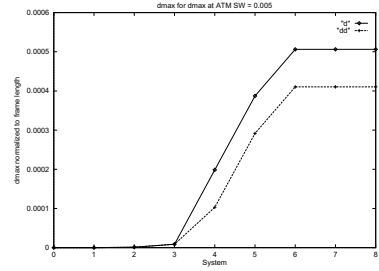
Figure 16 shows the loss ratio resulting from the delay constraint at the central ATM switch (System 4) and its mapping to the next layers.

The concatenation of all the components has the following parameters: In case of no loss, we have the same values as above. In case of loss,  $R = 139.83$  and  $T_0 = 0.11$ ;  $d_{max} = 0.11$  and  $B_{max} = 11027482.16$ . The cumulative values are  $d_{max} = 0.14$  and  $B_{max} = 14595026.77$ .

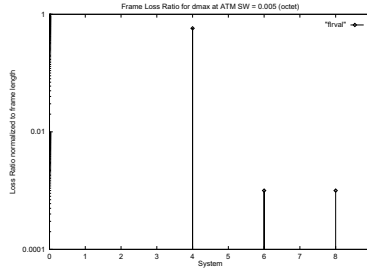




**Fig. 14.** Unconstrained vs delay-constrained maximum backlog



**Fig. 15.** Unconstrained vs delay-constrained maximum delay



**Fig. 16.** Delay-constrained loss ratio

## 5 Conclusion

In this work, we mapped loss and delay between single intserv IP flows and ATM, using NC, a min-plus algebra formulation. We specifically determined the maximum backlog and delay in the system components, their transition to a next layer as well as their formulation therein. As NC only applies to lossless systems, we extended the theory and introduced loss in two fashions: either by setting constraints on the buffer size or on the delay.

This work may be further extended to cover: i. the case of three IP QoS classes into one ATM central switch, i.e. intserv over ATM and ii. the case of three intserv IP classes into one IP class, i.e. intserv as a customer to diffserv, which consists mainly of investigating the mapping of QoS between a single flow versus an aggregation of flows.

## References

1. ISO/ITU-T, Quality of Service Framework, International Standard, December 1998.
2. ITU-T Recommendation I.350, General Aspects of Quality of Service and Network Performance in Digital Networks, including ISDNs, March, 1993.
3. ITU-T Recommendation I.356, -ISDN ATM Layer Cell Transfer Performances, October, 1996.

4. ITU-T Recommendation I.380, Internet Protocol (IP) Data Communication Service - IP Packet Transfer and Availability Performance Parameters, March, 1998.
5. V. Paxson, G. Almes, J. Mahdavi, M. Mathis, RFC 2330, Framework for IP Performance Metrics, June, 1998.
6. Jae-Il Jung, Quality of Service in Telecommunications Part I: Proposition of a QoS Framework and Its Applications to B-ISDN, IEEE Communications Magazine, August, 1996.
7. T. Chahed, S. Ben Fredj, C. Fayet, Native ATM versus IP over ATM Solutions : Comparative Studies, IEEE ATM'99 Workshop, Kochi City, May 1999.
8. T. Chahed, C. Fayet, G. Hébuterne, Framework for Translation of QoS and Performance Parameters between ATM and IP, ISCC'99, Red Sea, July 1999.
9. S. Shenker, J. Wroclawski, RFC 2215, General Characterization Parameters for Integrated Service Network Elements, September 1997.  
Specification of Guaranteed Quality of Service, September 1997.  
Controlled-Load Network Element Service, September 1997.
10. S. Blake, D. Black, M. Carlson, E. Davies, Z. Wang, W. Weiss, RFC 2475, An Architecture for Differentiated Services, December 1998.
11. E. Crawley, L. Berger, S. Berson, F. Baker, M. Borden, J. Kraw RFC 2382, A Framework for Integrated Services and RSVP over ATM, August 1998.
12. M. Garrett, M. Borden, RFC 2381, Interoperation of Controlled-Load Service and Guaranteed Service with ATM, August 1998.
13. J-Y. Le Boudec, Application of Network Calculus To Guaranteed Service Networks, IEEE Trans on Information theory, (44) 3, May 1998.
14. J-Y. Le Boudec, P. Thiran, Network Calculus viewed as a Min-Plus System Theory, SSC research report SSC/1998/016, EPFL, 1998.
15. R. L. Cruz, Quality of Service Guarantees in Virtual Circuit Switched Networks, IEEE JSAC, 1995.
16. A. Romanow, S. Floyd, Dynamics of TCP Traffic over ATM Networks, IEEE JSAC, V. 13 N. 4, May 1995, p. 633-641.
17. O. Elloumi, H. Afifi, Improving RED Algorithm Performance in ATM, IEEE Globecom'97, Phoenix, November 1997.
18. J. Roberts, U. Mocci, J. Virtamo (Eds), *Broadband Network Teletraffic*, Final Report of Action COST 242, Springer, 1996.

# A Self-Similar Point Process Through Fractal Construction

Manjunath A. Krishnam, Anand Venkatachalam, and Jeffrey M. Capone

Telecommunications Research Center (TRC)  
Department of Electrical Engineering  
Arizona State University, Tempe, AZ 85287-7206  
{manjunath, anandv, jcapone}@asu.edu

**Abstract.** A self-similar point process is developed by embedding a process with bursty behavior over timescales. This embedding is not arbitrary, but achieved through a model which itself has fractal patterns. This model decomposes the self-similar point process over its timescales in a manner that may be tractable for accurate characterization and control of packet traffic. The limiting behavior of the model is shown to possess the properties of a self-similar point process, namely, bursts of arrivals which have no (intrinsic) timescale. Furthermore, this model leads to efficient synthesis of such a process.

## 1 Introduction

In [1,2] the term self-similarity is used to describe the fractal (a form of invariance with respect to changes in scale) nature of a stochastic process in its distribution, more specifically,  $Y(t) =_d c^{-H} Y(ct)$ , where  $=_d$  is equality in distribution. Fractal (or self-similar) processes are among many possible models for generating long memory (long-range) dependent process. For example, (fractional) Brownian motion is a self-similar process and its increments may yield a long-range dependent process. Not every fractal process will give rise to long range dependent process, for example standard Brownian motion and its increments.

Long-range dependence can be observed in many ways. One of the most common ways is to examine the correlation function,  $\rho(k)$ . A long-range dependent process will have a correlation structure that decays to zero at a rate that is proportional to  $k^{-\alpha}$  where  $0 < \alpha < 1$ , so that  $\sum_{k=n}^{\infty} \rho(k) = \infty$ ,  $\forall n < \infty$ . A consequence of this correlation structure is that the variance of the sample mean of the process decays to zero at a rate slower than  $n^{-1}$ ,  $\lim_{n \rightarrow \infty} \frac{\text{var}[\sum_{i=1}^n X_i]}{n^{2-\alpha}} = K$  for  $0 < \alpha < 1$ , which has been typically referred to as asymptotic self-similarity [3,4,5,6].

Asymptotic self-similar traffic has been observed to occur in many communication networks [3,7,8]. This traffic tends to be so bursty, that even bursts are composed of bursts of bursts (a fractal “like” property). Several useful models have been proposed that capture this behavior, namely the  $M|G|\infty$  (with Pareto service times) model [9], the superposition of two state Markovian sources [10],

the mixture of exponentials to fit the long-tail distributions citeFeldman, the superposition of  $N$  *On-Off* processes with sub-exponential *On* periods [12,13], deterministic chaotic maps [14] and self-similar (fractal) point processes [1,15,4]. In each of the above models, it has been shown that the number of arrivals over an interval (number of busy servers in  $M|G|\infty$  model) all exhibit a long-range dependent correlation structure.

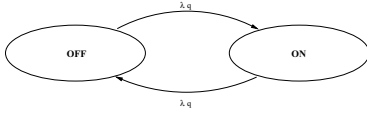
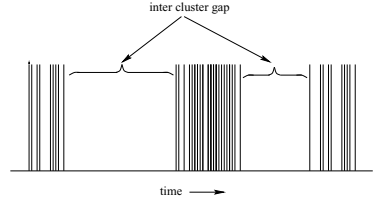
In this work, a self-similar (fractal) point process is developed. A self-similar point process, as defined by [1] is one whose inter-arrival times follow the “uniformly self-similar law of probability.” Mathematically,  $Pr(\frac{U}{\epsilon_1} > u | \frac{U}{\epsilon_1} > 1) = Pr(\frac{U}{\epsilon_2} > u | \frac{U}{\epsilon_2} > 1)$ , where  $U$  is the random variable corresponding to the inter-arrival times. The model developed in this work, is based on a fractal construction of a basic point process (cluster process), where clusters are embedded over an infinite number of timescales. This construction is the basis of a self-similar point process where points appear to be grouped in clusters, which in-turn are grouped in clusters of clusters, etc. [1]. This construction leads to bursts which have no (intrinsic) timescale. In addition, this model decomposes the self-similar point process in a manner that may be tractable for the accurate characterization and control of packet traffic. The model may also allow for further understanding the features of self-similar traffic and how they may impact network performance. Furthermore, this model leads to efficient synthesis of such a process.

## 2 Construction of a Self-Similar Point Process

In this section, a model for self-similar point processes is developed. The model is based on a fractal construction of a basic point (cluster) process, where clusters are embedded over all timescales.

### 2.1 Basic Process

Consider a process that generates Poisson arrivals with rate  $\lambda$ . After each arrival instant, a decision is made with probability  $p$  to continue generating arrivals with rate  $\lambda$  or with probability  $1 - p = q$  to turn off for a period of time. The number of arrivals,  $N$ , before entering an *Off* period is geometrically distributed with a mean  $q^{-1}$ . Thus, the time spent in an *On* period,  $\tau = \sum_{i=1}^N X_i$ , is a geometric sum of independent identically distributed (*i.i.d*) exponential random variables,  $X_i$ . It is shown in the Appendix, that  $\tau$  is an exponentially distributed random variable with parameter  $\lambda q$ . Therefore, if the *Off* periods are taken to be exponentially distributed, then an *On-Off* Markovian model, as shown in Fig. 1, may be used to describe this basic process. The timescale of this basic point process is  $\frac{1}{q}$ . A realization of the point process generated by this basic *On-Off* process is seen in Fig. 2. The exponential parameter of the *Off* period is also taken to be  $\lambda q$ . In addition, it is also assumed that an arrival is generated upon departing from an *Off* period. The inter-arrival time probability density function for this basic process is,  $f_{1X}(x) = f_{1X|on}(x)P_{on} +$

**Fig. 1.** *On-Off* Markovian model.**Fig. 2.** Point process generated by an *On-Off* Markovian model.

$f_{1X|off}(x)P_{off}$ , where  $f_{1X|on}(x)$  and  $f_{1X|off}(x)$  are the inter-arrival distributions while in the *On* and *Off* states, respectively.  $P_{on}$  is the probability that an inter-arrival time is drawn from the  $f_{X|on}(x)$  distribution.  $P_{on}$  may be viewed as,  $P_{on} = \frac{E[\# \text{ of arrivals during an on period over one cycle}]}{E[\text{total } \# \text{ of arrivals over one cycle}]} = \frac{1}{1+q}$ . Similarly  $P_{off} = \frac{q}{1+q}$ . The inter-arrival distribution of this process is,

$$f_{1X}(x) = \begin{cases} \frac{1}{1+q}\lambda e^{-\lambda x} + \frac{q}{1+q}\lambda q e^{-\lambda q x} & \text{for } x \geq 0 \\ 0 & \text{otherwise.} \end{cases} \quad (1)$$

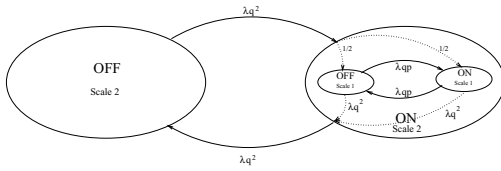
This basic model along with its properties are repeatedly used for the fractal construction of a self-similar point process.

## 2.2 Fractal Construction

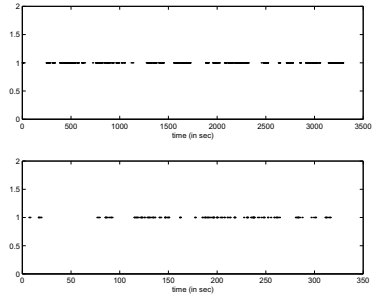
As it can be seen in Fig. 2, the basic process generates a burst of arrivals over one timescale. The burst lengths are exponentially distributed with parameter  $\lambda q$  and the idle periods are also exponentially distributed. To obtain burstiness on the next timescale, this basic process is embedded into the *On* state of another basic process. This two-scale process may be viewed on the next timescale as the basic process. However, an arrival on this next timescale represents a cluster on the lower timescale. Furthermore, the inter-cluster gaps, as seen in Fig. 2, are exponentially distributed with parameter  $\lambda q$ . Therefore, arrivals on this next timescale during an *On* period may also be viewed as occurring according to a Poisson process with rate  $\lambda q$ .

Again, upon each arrival (cluster at the lower timescale), a decision<sup>1</sup> is made with probability  $q$  to enter an idle period. Thus, the time spent in the *On* state at this next timescale is a geometric sum of *iid* exponential random variables (*On* and *Off* periods of the lower scale) with parameter  $\lambda q$ , and hence, the arrival *On* period is exponentially distributed with parameter  $\lambda q^2$ . The *Off* period at this next scale is also taken to be exponentially distributed with parameter  $\lambda q^2$ . When the process described above enters an *On* period, the state of the next lower scale process is selected with equal probability. A diagram for this two

<sup>1</sup> A decision is made at the beginning and ending of each arrival to maintain a geometric sum of *iid* exponential random variables.



**Fig. 3.** Embedding an *On-Off* model of one scale into another.



**Fig. 4.** Point process with one level of embedding and two time scales.

timescale process is shown in Fig. 3. The lower timescale is  $\frac{1}{q}$ , while the higher timescale is  $\frac{1}{q^2}$ .

During an *On* period of the higher timescale process, the lower timescale process alternates between *On* and *Off* periods, see Fig. 3. The number of times it alternates between the *On* and the *Off* periods is a geometric random variable with mean  $q^{-1}$ . Thus, there will be (on average)  $\frac{1}{2q}$  visits to the *Off* state of the lower timescale and  $\frac{1}{q}$  visits to the *On* state of the lower timescale. Each visit to the *On* state of the lower timescale generates (on average)  $\frac{1}{q}$  arrivals. Therefore, the probability that a given arrival occurs from leaving the off period of the higher timescale process is  $\frac{1}{\frac{1}{2q} + \frac{1}{q} + 1} = \frac{(2q)^2}{2+2q+(2q)^2}$ . Similarly, the probabilities that a given arrival occurs for the *Off* and *On* periods of the lower scale process are  $\frac{2q}{2+2q+(2q)^2}$  and  $\frac{2}{2+2q+(2q)^2}$ , respectively. The inter-arrival time distribution for this two-scale point process is,

$$f_{2X}(x) = \begin{cases} \frac{2\lambda e^{-\lambda x} + 2q\lambda q e^{-\lambda q x} + (2q)^2 \lambda q^2 e^{-\lambda q^2 x}}{2+2q+(2q)^2} & \text{for } x \geq 0 \\ 0 & \text{otherwise.} \end{cases}$$

This point process consists of clusters, which in-turn are composed of clusters of arrivals. This behavior is shown in a realization of this point process over two timescales in Fig. 4.

Now consider a point process constructed in a fashion as described above which contains  $n$  timescales. This process may be viewed as the basic process embedded in the *On* state of the  $n^{th}$  timescale. The time between visits to the *On* and *Off* periods in this  $n$  timescale process is exponentially distributed with parameter  $\lambda q^n$ . The diagram for the  $n$  timescale process is shown in Fig. 5. For each arrival generated from leaving an *Off* period at the highest timescale, (on average)  $\frac{1}{2q}$  arrivals will be generated from leaving an *Off* period at the next lower timescale,  $\frac{1}{2q^2}$  arrivals will be generated by the following *Off* period of the next timescale, etc. Continuing in this manner, for every event that occurs from leaving the *Off* period in the highest timescale, the average number of events that

**Table 1.** Probabilities and rate of arrivals occurring on a particular timescale.

State	Time Scale	Probability	Average arrival rate
<i>off</i>	$q^{-n}$	$\frac{(2q)^n}{2+\sum_{i=1}^n (2q)^i}$	$\lambda q^n$
$\vdots$	$\vdots$	$\vdots$	$\vdots$
<i>off</i>	$q^{-j}$	$\frac{(2q)^j}{2+\sum_{i=1}^n (2q)^i}$	$\lambda q^j$
$\vdots$	$\vdots$	$\vdots$	$\vdots$
<i>off</i>	$q^{-1}$	$\frac{2q}{2+\sum_{i=1}^n (2q)^i}$	$\lambda q$
<i>on</i>	$q^{-1}$	$\frac{2}{2+\sum_{i=1}^n (2q)^i}$	$\lambda$

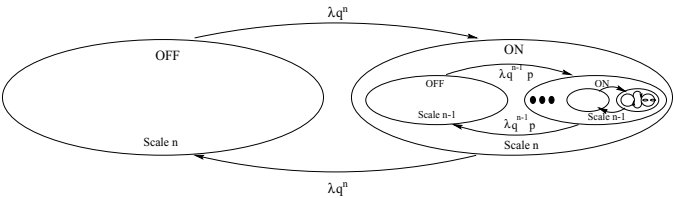
occur due to the process being in the *Off* states of any timescale can be identified. Table 1 contains the probability of an arrival generated from a particular state.

Given that the process is in the *Off* period at timescale  $i$ , the time spent in this period is exponential with parameter  $\lambda q^i$ . Therefore, the inter-arrival time distribution of the overall point process is,

$$f_{nX}(x) = \begin{cases} \frac{2\lambda e^{-\lambda x} + \sum_{i=1}^n (2q)^i \lambda q^i e^{-\lambda q^i x}}{2 + \sum_{i=1}^n (2q)^i} & \forall x \geq 0 \\ 0 & \text{otherwise.} \end{cases} \quad (2)$$

It is easily verified that  $\int_0^\infty f_{nX}(x) dx = 1$ . As the number of timescales approach infinity, interesting properties of the distribution of the time between clusters (inter-cluster gap) develop. Let  $f_X(x) \equiv \lim_{n \rightarrow \infty} f_{nX}(x)$ , which may be shown to be uniformly convergent (see Section 3.1).

**Distribution of Inter-cluster Times** A cluster at the  $(i-1)$  timescale corresponds to an arrival at timescale  $i$ . If a cluster can be assumed to be a point (by compressing time by a factor of  $q^{i-1}$ ) then the model may be assumed to start from scale  $i$  and has  $(n-i)$  timescales. Therefore, the inter-cluster times



**Fig. 5.** Model to construct fractal point process over  $n$  timescales.

$Y$  have a probability density function,

$$f_{nY}(x) = \begin{cases} \frac{2\lambda q^{i-1} e^{-\lambda q^{i-1} x} + \sum_{j=1}^{n-i+1} (2q)^j \lambda q^{i+j-1} e^{-\lambda q^{i+j-1} x}}{2 + \sum_{j=1}^{n-i+1} (2q)^j} & \forall x \geq 0 \\ 0 & \text{otherwise.} \end{cases} \quad (3)$$

Scaling  $Y$  by  $q^{i-1}$ , the new random variable,  $Y' = q^{i-1}Y$ , has a probability density function,

$$f_{lY'}(x) = \begin{cases} \frac{2\lambda e^{-\lambda x} + \sum_{j=1}^l (2q)^j \lambda q^j e^{-\lambda q^j x}}{2 + \sum_{j=1}^l (2q)^j} & \forall x \geq 0 \\ 0 & \text{otherwise,} \end{cases}$$

where  $l = n - i + 1$ . Let  $f_{Y'}(x) \equiv \lim_{l \rightarrow \infty} f_{lY'}(x)$ , which is also uniformly convergent.

Consider the complementary cumulative distribution function (ccdf) for  $X$  and  $Y$ . It is seen that

$$P(X > u) = \int_u^\infty f_X(x) dx = \int_u^\infty f_{Y'}(x) dx = P(Y' > u).$$

(Note that these are the same values of  $u$ , but on different timescales). Therefore, the probability that an inter-cluster time is greater than  $u$  of its time units is equal to the probability that the inter-arrival time of points is greater than  $u$  of its time units. Since,  $n \rightarrow \infty$  ( $l \rightarrow \infty$ ), this result holds for all finite timescales. Thus, a burst on any timescale will be statistically the same, a fundamental characteristic of a fractal point process.

### 3 Analysis of the Model

The model developed had no strict restrictions on the parameters used to describe it. The only conditions on the parameters were  $\lambda > 0$  and  $0 < q < 1$ . The density function has to be investigated further to find whether any more restrictions on the model parameters are needed to make the probability density function a valid density function, as  $n \rightarrow \infty$ .

#### 3.1 Uniform Convergence of the Probability Density Function

Using the model described in Section 2, points (arrivals) can be generated with an inter-arrival distribution given in (2). Several characteristics are observed in this process as the number of timescales embeddings approach infinity, namely, the inter-cluster time distributions were statistically the same and only differed in timescale. Convergence of this distribution  $f_X(x)$  for any timescale is studied in this section. Uniform convergence of  $f_X(x) = \lim_{n \rightarrow \infty} f_{nX}(x)$  can be shown by applying the Weierstrass' M-test [16], as follows. For  $x < 0$ , the probability density function is always zero. For  $x \geq 0$ , the probability density function can



be analyzed by considering it to be the quotient of two series. The probability density function  $f_X(x)$ , for  $x \geq 0$ , is,

$$f_X(x) = \frac{2\lambda e^{-\lambda x} + \sum_{i=1}^{\infty} (2q)^i \lambda q^i e^{-\lambda q^i x}}{2 + \sum_{i=1}^{\infty} (2q)^i}.$$

When  $q \in (0, 0.5)$ , the denominator in the above expression evaluates to  $\frac{2-2q}{1-2q}$ . The infinite series in the numerator is  $\lambda[2e^{-\lambda x} + \sum_{i=1}^{\infty} (2q^2)^i e^{-\lambda q^i x}]$ . Let  $M_n = (2q^2)^n$  for  $n > 0$  and  $M_n = 2$  for  $n = 0$ . Then,  $\sum_{n=0}^{\infty} M_n$  converges absolutely if  $2q^2 < 1$ . For  $q \in (0, 0.5)$ ,  $2q^2 \leq 1$  and  $\sum_{n=0}^{\infty} M_n$  converges to  $\frac{2-2q^2}{1-2q^2}$ . If  $g_n(x)$  is defined as

$$g_n(x) = \begin{cases} 2e^{-\lambda x} & \text{for } n = 0 \\ (2q^2)^n e^{-\lambda q^n x} & \text{for } n > 0, \end{cases}$$

then it is seen that for all  $x > 0$  and  $\lambda > 0$ ,  $|g_n(x)| \leq M_n, \forall n > 0$  and therefore,  $Nr\{f_X(x)\}$  converges uniformly  $\forall x \geq 0$ . Thus, it is seen that  $f_X(x)$  converges uniformly  $\forall x \geq 0$ , when the generation process consists of infinite number of timescale embeddings.

### 3.2 Moments of the Density Function

The mean inter-arrival time for this process with  $0 \leq q \leq 0.5$  may be evaluated as,

$$\begin{aligned} E[x] &= \int_0^{\infty} \frac{2\lambda x e^{-\lambda x} + \sum_{i=1}^n (2q)^i \lambda q^i x e^{-\lambda q^i x}}{2 + \sum_{i=1}^n (2q)^i} dx \\ &= \frac{\frac{1}{\lambda}(2 + \sum_{i=1}^n 2^i)}{2 + \sum_{i=1}^n (2q)^i}. \end{aligned} \quad (4)$$

As  $n \rightarrow \infty$ ,

$$\begin{aligned} E[X] &= \frac{\frac{1}{\lambda}(2 + \sum_{i=1}^{\infty} 2^i)}{2 + \sum_{i=1}^{\infty} (2q)^i} \\ &= \infty. \end{aligned} \quad (5)$$

Therefore, as the number of timescales embedded in the process approaches infinity, the mean inter-arrival time approaches infinity and the mean arrival rate  $E[X]^{-1}$  approaches zero. This result is another fundamental and necessary characteristic of a self-similar point process [1].

### 3.3 Behavior of the Density Function Compared to a Pareto Distribution

A point process can be called fractal if the following phenomenon is observed. On a particular timescale, the point process looks bursty, with a lot of arrivals

clustered around one point. If the time axis is dilated about this point and the cluster is closely observed, the cluster looks bursty again leading to the formation of clusters over the new timescale. If on continuous dilations of the timescale about a point where the arrivals appear to be clustered, results in clustering of points again over several timescales, the point process is said to exhibit fractal properties. Alternatively, it can be explained as follows. The distribution of the inter-arrival time on a particular timescale and the distribution of the inter-arrival time on another timescale, differ only in scale. In [1], it is shown that the only distribution that satisfies the above condition, is the Pareto distribution. A Pareto distribution can be described by its probability density function,

$$f_X(x) = \begin{cases} K_\epsilon x^{-(\gamma+1)} & x \geq \epsilon \\ 0 & \text{otherwise,} \end{cases} \quad (6)$$

where  $K_\epsilon$  is a normalizing constant to make  $f_X(x)$  a valid density function.

A Pareto distribution may be expressed as a weighted sum of exponentials by considering the Gamma function. The Gamma function is,  $\Gamma(D) = \int_0^\infty e^{-y} y^{D-1} dy$ . Allowing  $y = bx$ ,  $dy = xdb$  and observing that the limits remain the same, the following equation is obtained,  $x^{-D} = \int_0^\infty e^{-bx} b^{D-1} db \Gamma(D)^{-1}$ . Approximating this integral by a summation,  $x^{-D}$  may be written as a sum of weighted exponentials. This approach may be used to relate the fractal parameter  $D$ , to the parameters of this model. If the probability density function of the inter-arrival times for a point process generated by this model behaves as  $x^{-D}$ , a relationship between the model parameters and the fractal  $D$  may be obtained. Approximating the integral,

$$x^{-D} = \Gamma(D)^{-1} \sum_{m=0}^{\infty} e^{-b_m x} b_m^{D-1} (b_m - b_{m+1}).$$

Using the substitution  $b_m = \lambda q^m$ , the above equation becomes,

$$\begin{aligned} x^{-D} &= \Gamma(D)^{-1} \sum_{m=0}^{\infty} e^{-\lambda q^m x} (\lambda q^m)^{D-1} (\lambda q^m) (1 - q) \\ &= G(D) \sum_{m=0}^{\infty} \lambda (q^m)^D e^{-\lambda q^m x}, \end{aligned} \quad (7)$$

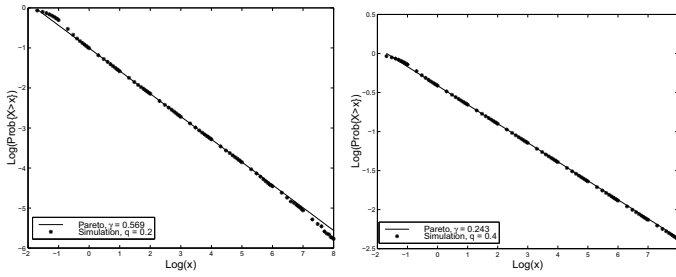
where  $G(D) = \Gamma(D)^{-1} \lambda^{D-1} (1 - q)$ .

The probability density function can be written as,

$$f_X(x) = \frac{2 - 2q}{1 - 2q} [\lambda e^{-\lambda x} + \sum_{m=0}^{\infty} \lambda (2q^2)^m e^{-\lambda q^m x}]. \quad (8)$$

For large values of  $x$  the contribution of the initial terms in (7) and (8) are negligible. Comparing the remaining terms of (7) to those of (8), it is seen that  $q^{mD} = (2q^2)^m$  or,

$$D = \frac{\log 2}{\log q} + 2, \quad (9)$$



**Fig. 6.** Loglog plot of ccdf of the simulated data vs. Pareto distribution.

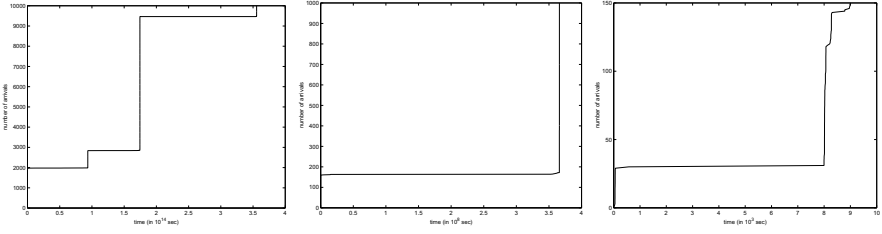
and  $\gamma = (D - 1)$ , where  $\gamma$  is the fractal exponent of the Pareto distribution in (6). It is seen that for  $0 < q < 0.5$ , then  $0 < \gamma < 1$ , which is the range for which the point process in [1] behaves as a *fractal point process*.

## 4 Results

In this section, the simulations using the above described model are compared to the theoretical Paretian distributions. Fig. 6 compares the complementary cumulative distribution function of the point process simulated to that of the corresponding Pareto distribution. It may be seen from these plots that the probability density function of the point process generated by this model approaches a Paretian distribution. Fig. 7 shows the counting process of a single realization over three different time scales. The first plot contains the first 10000 points of the simulation. The plot shows the strong clustering of data. On zooming in, on the first cluster, the second plot is obtained, which again shows strong clustering of data. On further zooming in, the third plot is obtained, which shows strong clustering of points again. The clustering of clusters and the reduction of a cluster into a point on higher timescales may be illustrated in this figure.

## 5 Conclusion

This model was developed based on the fact of clustering of arrivals within clusters as in present day telecommunication traffic. The concept of embedding an *On-Off* process into another *On-Off* process gives more structure to the understanding of these clusters. In [1], it is shown that if the inter-arrival times follow a Paretian distribution, the inter-cluster time distribution differ from the inter-arrival time distribution only in scale. The analysis of the model developed , shows that the distribution of the inter-cluster time and that of the inter-arrival time of the point process, differ only in scale. In addition, the model allows for efficient synthesis of the process. The simulation results show that the complementary cumulative distribution function of the inter-arrival times behaves as a Pareto distribution.



**Fig. 7.** Counting process showing strong clustering of arrivals over different timescales.

## A Geometric Sum of Exponential Random Variables

**Theorem 1.** *If  $S = \sum_{i=1}^N X_i$ , where  $X_i$ s are independent identically distributed exponential random variables with parameter  $\lambda$  and  $N$  is a geometric random variable with parameter  $q$ , then  $S$  is an exponential random variable with parameter  $\lambda q$ .*

*Proof:*

The distribution of  $S$  is found by first conditioning on  $N = n$ ,

$$f_{S|N=n}(s) = \begin{cases} \frac{\lambda^n s^{n-1}}{(n-1)!} e^{-\lambda s} & \forall s \geq 0 \\ 0 & \text{otherwise.} \end{cases}$$

Taking expectation with respect to  $N$ ,

$$\begin{aligned} f_S(s) &= \sum_{n=1}^{\infty} \frac{\lambda^n s^{n-1} p^{n-1} q e^{-\lambda s}}{(n-1)!} \\ &= \lambda q e^{-\lambda s} e^{-\lambda s p} \\ &= \lambda q e^{-\lambda s q}. \end{aligned}$$

Thus, the probability density function of  $S$  is,

$$f_S(s) = \begin{cases} \lambda q e^{-\lambda q s} & \forall s \geq 0 \\ 0 & \text{otherwise.} \end{cases}$$

## B Algorithm for Generating a Self-Similar Point Process

The algorithm used to synthesize the fractal point process of Section 2, is described in this section. This algorithm assumes embedding of *On-Off* processes over infinite timescales. The other parameters to be specified are  $\lambda$  and  $\gamma$ .  $\lambda$  is determined by the value of time above which the probability density function or the complementary cumulative distribution function behaves as a Pareto distribution. This may be viewed as setting the value of  $\epsilon$  in (??).  $\gamma$  is the parameter of the Pareto distribution.

This algorithm consists of four functions. The variables *currentState* and *q* are used in the algorithm. *currentState* corresponds to the state and the timescale in which the generation process is residing. *currentState* = 1 corresponds to the *On* state of the lowest timescale. *currentState* = 2 corresponds to the *Off* state of the lowest timescale. Continuing similarly *currentState* = *n* corresponds to the *Off* state of the  $(n - 1)^{th}$  timescale. The value of *q* depends on the exponent of the Pareto distribution, to which the probability density function of the inter arrival time approaches. It is calculated from the value of  $\gamma$ , using relation (9). A random initial state is chosen according to the probabilities given in Table 1. The process starts off initially with *currentState* set to this number. The second function NEXTSTATE computes the state to which the generation process will jump next. This function uses two other functions SUCCESS and FAILURE. SUCCESS and FAILURE describe the sequence of events that occur when the process has to leave any particular state.

GENERATE( $\lambda, \gamma$ )

1.  $q \leftarrow 2^{-\frac{1}{1-\gamma}}$
2. **initialize** *currentState*
3. *time*  $\leftarrow 0$
4. **while**(1)
5.     **do** generate exponential random number, *exp*, with parameter  $\lambda q^{currentState-1}$
6.     *time*  $\leftarrow time + exp$
7.     Generate an arrival at *time* seconds
8.     *currentState*  $\leftarrow$  NEXTSTATE(*currentState*)

NEXTSTATE(*currentState*)

1. Generate a uniform random number *unirand*  $\in (0, 1)$ .
2. **if** (*unirand* > *q*)
3.     *currentState*  $\leftarrow$  SUCCESS(*currentState*)
4. **else**
5.     *currentState*  $\leftarrow$  FAILURE(*currentState*)
6. **return** *currentState*

SUCCESS(*currentState*)

1. **if** (*currentState* = 1)
2.     **return** *currentState*
3. **else**
4.     *currentState*  $\leftarrow currentState - 1$
5.     Generate uniform random number *unirand*  $\in (0, 1)$
6.     **if** (*unirand* < 0.5)
7.         **return** *currentState*
8.     **else**
9.         **return** SUCCESS(*currentState*)

FAILURE(*currentState*)

1.  $currentState \leftarrow currentState + 1$
2. Generate uniform random number  $unirand \in (0, 1)$
3. **if** ( $unirand > q$ )
4.     **return**  $currentState$
5. **else**
6.     **return** FAILURE( $currentState$ )

## References

1. B. B. Mandelbrot, "Self-similar error clusters in communication systems and the concept of conditional stationarity," *IEEE Transactions on Communication Technology*, vol. COM-13, pp. 71–90, March 1965.
2. J. Beran, *Statistics for Long-Memory Process*. New York: Chapman and Hall, 1994.
3. M. E. Crovella and M. S. Taqqu, "Estimating the heavy tail index from scaling properties," *Methodology and Computing in Applied Probability*, vol. 1, no. 1, 1999 to appear.
4. S. B. Lowen and M. C. Teich, "Fractal renewal processes generate  $1/f$  noise," *Physics Review*, vol. E, no. 47, pp. 992–1001, 1993.
5. B. K. Ryu and S. B. Lowen, "Point Process Approaches to the Modeling and Analysis of Self-Similar Traffic – Part I: Model Construction," in *Proc. IEEE INFOCOM'96*, pp. 1468–1475, March 1996.
6. B. Tsybakov and N. D. Georganas, "Self-similar processes in Communications Networks," *IEEE Transactions on Information Theory*, vol. 44, pp. 1713–1725, September 1998.
7. W. Leland, M. Taqqu, W. Willinger, and D. Wilson, "On the self-similar nature of Ethernet traffic.," *IEEE/ACM transactions on Networking*, vol. 2, pp. 1–15, February 1994.
8. V. Paxson and S. Floyd, "Wide-area traffic: the failure of Poisson Modeling," in *Proc. Sigcomm'94*, pp. 257–268, October 1994.
9. M. Parulekar and A. Makowski, "Tail Probabilities for a multiplexer with self-similar traffic," in *Proc. IEEE INFOCOM'96*, pp. 1452–1459, March 1996.
10. A. T. Andersen and B. F. Nielsen, "An application of superposition of two state Markovian sources to the modelling of self-similar behavior," in *Proc. IEEE INFOCOM'97*, 1997.
11. A. Feldmann and W. Whitt, "Fitting Mixture of Exponentials to Long-Tail Distributions to Analyze Network Performance Models," *Performance Evaluation*, vol. 31, pp. 245–279, January 1998.
12. N. Likhanov, B. Tsybakov, and N. D. Georganas, "Analysis of an ATM buffer with self-similar ("fractal") input traffic," in *Proc. IEEE INFOCOM'95*, 1995.
13. P. R. Jelenkovic and A. A. Lazar, "Asymptotic Results for Multiplexing Subexponential On-Off Processes," *Advances in Applied Probability*, vol. 31, no. 2, 1999 to appear.
14. A. Erramilli, R. P. Singh, and P. Pruthi, "Modeling Packet Traffic with Chaotic Maps," *Royal Institute of Technology, ISRN, KTH/IT/R-94/18*, August 1994.
15. W. M. Lam and G. W. Wornell, "Multiscale Representation and Estimation of Fractal Point Processes," *IEEE Transactions on Signal Processing*, vol. 43, pp. 2606–2617, November 1995.
16. W. Rudin, *Principles of Mathematical Analysis*. McGraw Hill, 1976.

# Tail Transitions in Queues with Long Range Dependent Input

Tim Daniëls and Chris Blondia

University of Antwerp, Dept. Mathematics and Computer Science,  
Universiteitsplein 1, B-2610 Antwerpen  
blondia@uia.ua.ac.be,  
<http://win-www.uia.ac.be/u/pats>

**Abstract.** This paper studies the tail of the buffer occupancy distribution of a queueing system with  $c$  parallel deterministic servers, infinite buffer capacity and an input process with consists of a superposition of a long range dependent on-off source and a batch renewal process. In particular, we investigate the decay of the tail for different values of  $c$  and for different compositions of the traffic mix. It is shown that for  $c = 1$  (i.e. a single server system), the tail has a power law decay, while for  $c > 1$ , different cases may be distinguished: if the arrival rate of the background traffic is larger than  $c - 1$ , then the tail probabilities decay according to a power law, while for the other case, the decay is exponential.

## 1 Introduction

In the past years, traffic measurements on packet networks (e.g. Ethernet LAN traffic [11], VBR video streams over ATM [8]) have shown that the autocorrelation function of the number of arrivals in a time interval does not decrease exponentially in time, but according to a power-law. This type of traffic is referred to as long range dependent traffic (LRD), in contrast with the more classical short range dependent traffic (SRD) (e.g. Markovian models). The autocorrelation structure of the arrival process has a major impact on the buffer occupancy of a queueing system with this process as input. For Markovian traffic, the tail of the queue length distribution decays exponentially, while for LRD input traffic the tail of the buffer distribution decays according to a power law (see e.g. [5], [7], [9], [10]). In [2], an exact formula for the buffer asymptotics of a discrete-time queue with an LRD M/G/ $\infty$  input process was obtained. The M/G/ $\infty$  is defined as a process where trains arrive in a time slot according to a Poisson process with rate  $\lambda$ . Each train consists of a number of back-to-back arrivals with length  $\tau_A$ , with distribution given by  $\mathbf{P}\{\tau_A = k\} \sim ak^{-s}$ , with  $2 < s < 3$  and  $a > 0$ . The buffer asymptotics are then determined by

$$\mathbf{P}\{q > k\} \sim \frac{\lambda a \rho^{s-2}}{(s-2)(s-1)(1-\rho)} k^{2-s}, \quad (1)$$

where  $q$  denotes the buffer occupancy and  $\rho$  is the load of the system. In this paper we consider a queueing system where the input process consists

of a mix of LRD and SRD traffic. The LRD component is an on-off source with geometrically distributed off-period and on-periods which have a Pareto-like distribution. The SRD component consists of a batch renewal process. To model the difference in speed of the input and output lines, we assume that this traffic mix is fed to a multi-server queue with  $c$  parallel servers. The aim of the paper is to investigate the tail of the buffer occupancy distribution of this (LRD+SRD)/D/ $c$  queue for different compositions of the traffic mix and different values of  $c$ . We show that when  $c = 1$ , the tail has a power law decay, while for  $c > 1$ , different cases may be distinguished: if the arrival rate of the background traffic is larger than  $c - 1$ , then the tail probabilities decay according to a power law, while for the other case, i.e. the arrival rate of the background traffic is less than  $c - 1$ , the decay is exponential. With respect to the transition point (i.e. when the arrival rate of the background traffic is exactly  $c - 1$ ), no conclusions can be drawn.

These results give some insight about the influence of spacing cells of an LRD traffic stream on the queueing behavior. The number of servers  $c$  is a measure for the spacing distance. The results show that by spacing an LRD traffic stream and mixing with SRD traffic, a tail which decays according to a power law may be changed into an exponential decay.

## 2 The Queueing System

The queueing system that is considered consists of a multiplexer with  $c$  servers each having a deterministic service time of one time slot, an infinite capacity buffer and input traffic which is the superposition of an on-off source and so-called background traffic. The on-off source has geometrically distributed off-periods and on-periods which have a Pareto-like distribution. Let  $\tau_A$  be the duration of the on-periods and let  $a_k = \mathbf{P}\{\tau_A = k\} \sim ak^{-s}$ , with  $2 < s < 3$  and  $a > 0$ . For these parameters the on-off source generates LRD traffic. Indeed, if  $X_k$  denotes the number of arrivals generated by the on-off source in slot  $k$ , then the results presented in [15] imply that

$$\text{Var}(X_1 + \dots + X_n) \sim \frac{\mathbf{E}[\tau_B]^2}{(\mathbf{E}[\tau_A] + \mathbf{E}[\tau_B])^3} \frac{a}{(4-s)(3-s)(s-1)} n^{4-s}. \quad (2)$$

Hence by [14], the source generates LRD traffic with Hurst parameter  $H = (4-s)/2$ . The generating function associated with the off-periods is given by

$$\frac{(1-\beta)z}{1-\beta z}. \quad (3)$$

Let  $A(z) = \sum_{k=1}^{\infty} a_k z^k$ . The background traffic generates  $X_n$  arrivals in slot  $n$ , with  $\{X_n, n \geq 1\}$  a sequence of i.i.d. random variables. The corresponding generating function is denoted by  $\phi(z)$ .

In what follows, we describe the input traffic using a matrix-analytic notation. The LRD on-off source can be considered as a D-BMAP (see [1]) as follows. Let



$p_n^a$  be the probability that the LRD source is in the  $(n+1)^{th}$  slot of an on-period given that it was in the  $n^{th}$  slot of this on-period in the previous slot. Then

$$p_n^a = \frac{1 - \sum_{i=1}^n a_i}{1 - \sum_{i=1}^{n-1} a_i}$$

Letting the first state of the LRD source be the off-state, then the generating function defining the transitions of the corresponding D-BMAP, is given by

$$\mathbf{D}(z) = \begin{pmatrix} \beta & (1-\beta)z & 0 & 0 & 0 & \dots \\ 1-p_1^a & 0 & p_1^a z & 0 & 0 & \dots \\ 1-p_2^a & 0 & 0 & p_2^a z & 0 & \dots \\ 1-p_3^a & 0 & 0 & 0 & p_3^a z & \dots \\ \vdots & \vdots & \vdots & \vdots & \vdots & \ddots \end{pmatrix}.$$

Remark that  $\mathbf{D}(z)$  is an infinite matrix. By the special structure of  $\mathbf{D}(z)$  it is possible to obtain an implicit equation for its Perron-Frobenius eigenvalue  $\lambda(z)$ :

$$\lambda(z) = \beta + (1-\beta)A\left(\frac{z}{\lambda(z)}\right).$$

The generating function of the D-BMAP describing the superposition of the on-off source and the background traffic is given by  $\phi(z)\mathbf{D}(z)$  with mean arrival rate

$$\rho = \phi'(1) + \frac{(1-\beta)A'(1)}{(1-\beta)A'(1) + 1}.$$

### 3 The Tail Probabilities

Consider the (LRD+SRD)/D/ $c$  queue defined in the previous section. We consider two cases, namely  $c = 1$  and  $c > 1$ .

#### 3.1 The Tail Probabilities When $c = 1$

Since there is only one server, the queueing model is a simple D-BMAP/D/1 queue. Define the generating function of the buffer occupancy as

$$Q(z) \stackrel{\text{def}}{=} \sum_{k=0}^{\infty} q_k z^k,$$

with  $q_k$  the steady-state probability of having  $k$  customers in this system. Clearly  $Q(z) = \mathbf{X}(z)\mathbf{e}$ , with  $\mathbf{X}(z)$  given by

$$\mathbf{X}(z) = (z-1)\mathbf{x}_0\phi(z)\mathbf{D}(z)(z\mathbf{I} - \phi(z)\mathbf{D}(z))^{-1}.$$

This expression is the Pollachek-Kinchin equation for the D-BMAP/D/1 queue, with the input determined by the generating function  $\phi(z)\mathbf{D}(z)$ . Since there is

only one off-state, the vector  $\mathbf{x}_0$  is given by  $\mathbf{x}_0 = (1 - \rho \ 0 \ 0 \ \dots)$ . The structure of  $\mathbf{D}(z)$  allows us to compute the vector

$$\mathbf{w}(z) \stackrel{\text{def}}{=} \mathbf{x}_0 \phi(z) \mathbf{D}(z) (z \mathbf{I} - \phi(z) \mathbf{D}(z))^{-1}, \quad (4)$$

with  $\mathbf{w}(z) = (w_0(z) \ w_1(z) \ \dots)$ . Some straightforward algebra leads to

$$w_0(z) = \frac{\beta \phi(z) z + (1 - \beta) \phi(z) A(\phi(z)) z}{z^2 - \beta z \phi(z) - A(\phi(z)) (1 - \beta) z \phi(z)},$$

$$w_1(z) = \frac{(1 - \beta) z^2}{z^2 - \beta z \phi(z) - A(\phi(z)) (1 - \beta) z \phi(z)},$$

and for  $k \geq 2$ ,

$$w_k(z) = \left[ 1 - \sum_{i=1}^{k-1} a_i \right] \phi(z)^k w_1(z). \quad (5)$$

Since  $Q(z) = \mathbf{X}(z) \mathbf{e} = (1 - \rho)(z - 1) \mathbf{w}(z) \mathbf{e}$ , it follows that

$$Q(z) = (1 - \rho)(z - 1) \left[ w_0(z) + w_1(z) \frac{1 - A(\phi(z))}{1 - \phi(z)} \right]. \quad (6)$$

By calculating  $Q'(1)$  we obtain

$$\mathbf{E}[q] = \sum_{k=0}^{\infty} \mathbf{P}\{q > k\} = \frac{A''(1)}{2} \frac{1}{\mathbf{E}[\tau_A] + \mathbf{E}[\tau_B]} \left( \frac{\phi'(1)^2}{1 - \rho} + \phi'(1) \right) + \Omega,$$

with  $\Omega$  denoting a term involving the finite quantities  $\beta$ ,  $\phi'(1)$ ,  $\phi''(1)$  and  $A'(1)$  (we suppose that all the moments of the background traffic exist). Since  $\mathbf{E}[\tau_A^2] = \infty$ ,  $Q'(z)$  diverges when  $z$  approaches 1.

The Tauberian Theorem for power series (see e.g. [6]) plays a central role in the derivation of an expression for the tail probabilities of the buffer occupancy.

**Theorem 1 (Tauberian theorem for power series).** *Let  $p_k \geq 0$  and suppose that*

$$P(z) = \sum_{k=0}^{\infty} p_k z^k$$

*converges for  $0 \leq z < 1$ . If  $L$  is a constant and  $0 \leq \sigma < \infty$ , then the following relations are equivalent:*

$$P(z) \sim \frac{L}{(1 - z)^\sigma} \text{ for } z \rightarrow 1 -$$

$$p_0 + p_1 + \dots + p_{n-1} \sim \frac{L}{\Gamma(\sigma + 1)} n^\sigma \text{ for } n \rightarrow \infty.$$

This theorem leads to an expression for the behaviour of  $Q'(z)$  near 1.

**Lemma 1.** *The behaviour of  $Q'(z)$  near 1 is given by*

$$Q'(z) \sim \frac{L}{(1-z)^{3-s}} \text{ for } z \rightarrow 1-, \quad (7)$$

with

$$L = \frac{a\Gamma(3-s)}{s-1} \frac{1}{\mathbf{E}[\tau_A] + \mathbf{E}[\tau_B]} \left( \frac{\phi'(1)^{s-1}}{1-\rho} + \phi'(1)^{s-2} \right).$$

*Proof.* To prove the statement (7) we rewrite  $Q(z)$ :

$$Q(z) = (1-\rho)(f_0(z) + f_1(z)),$$

with

$$f_0(z) = (z-1) \frac{t(z)}{z^2 - t(z)}, f_1(z) = (z-1) \frac{(1-\beta)z^2}{z^2 - t(z)} \frac{1 - A(\phi(z))}{1 - \phi(z)}, \quad (8)$$

$$t(z) = \beta z \phi(z) + (1-\beta)z \phi(z)A(\phi(z)). \quad (9)$$

Let us focus on the behaviour of  $f'_0(z)$  as the function  $f'_1(z)$  can be analysed in a similar way. A combination of the results for  $f'_0(z)$  and  $f'_1(z)$  leads to (7). Clearly

$$f'_0(z) = -\frac{(t(z)-z)^2}{(z^2-t(z))^2} + z^2 \frac{(z-1)^2 \sum_{k=0}^{\infty} (k+1)T_{k+2}z^k}{(z^2-t(z))^2},$$

with  $T_k = \sum_{j \geq k} t_j$ . Since the first term of  $f'_0(z)$  does not diverge for  $z \rightarrow 1-$ , we only need to take the second term into account. First we determine the asymptotic behaviour of  $T_k$  by applying Theorem 1 to  $t''(z)$ . Observe that

$$t''(z) \sim (1-\beta)z\phi(z) \frac{d^2}{dz^2} [A(\phi(z))] \text{ for } z \rightarrow 1-.$$

Let  $\sigma = 3-s$ . We have

$$\begin{aligned} & \lim_{z \rightarrow 1-} (1-z)^\sigma \frac{d^2}{dz^2} [A(\phi(z))] \\ &= \lim_{z \rightarrow 1-} \left( \frac{1-z}{1-\phi(z)} \right)^\sigma (1-\phi(z))^\sigma A''(\phi(z)) (\phi'(z))^2 \\ &= \lim_{y \rightarrow 1-} (1-y)^\sigma A''(y) (\phi'(1))^{2-\sigma}. \end{aligned}$$

Applying Theorem 1 twice,

$$A''(y) \sim \frac{1}{(1-y)^\sigma} a\Gamma(\sigma),$$

$$\sum_{k=0}^n k(k-1)t_k \sim \frac{a\Gamma(\sigma)}{\Gamma(\sigma+1)} n^\sigma (\phi'(1))^{2-\sigma} (1-\beta),$$

we obtain

$$T_k \sim \frac{a'}{\phi} (1)^{s-1} (1-\beta) k^{1-s}.$$

One more application of Theorem 1 results in

$$f'_0(z) \sim \frac{1}{(2-t'(1))^2} \frac{a\Gamma(3-s)}{(s-1)\Gamma(4-s)} (1-z)^{s-3}.$$

Furthermore observe that

$$(1-\rho) = \frac{2-t'(1)}{(1-\beta)A'(1)+1},$$

hence

$$f'_0(z) \sim \frac{1}{(1-\rho)^2((1-\beta)A'(1)+1)^2} \frac{a\Gamma(3-s)}{(s-1)\Gamma(4-s)} (1-z)^{s-3}.$$

Using the Tauberian theorem for power series, the asymptotic behaviour of the coefficients of  $Q'(z) = \sum_k kq_k z^{k-1}$  follows from (7). Applying this theorem to  $Q'(z)$  results in

$$q_1 + 2q_2 + \dots + nq_n \sim \frac{L}{\Gamma(4-s)} n^{3-s}. \quad (10)$$

Using [12, 3.3 (c), pg. 59], we obtain

$$\sum_{j>k} q_j \sim \frac{L}{\Gamma(4-s)} \frac{3-s}{s-2} k^{2-s},$$

or

$$\mathbf{P}\{q > k\} \sim \frac{a}{(s-1)(s-2)} \frac{1}{\mathbf{E}[\tau_A] + \mathbf{E}[\tau_B]} \left( \frac{\phi'(1)^{s-1}}{1-\rho} + \phi'(1)^{s-2} \right) k^{2-s}. \quad (11)$$

*Numerical example .*

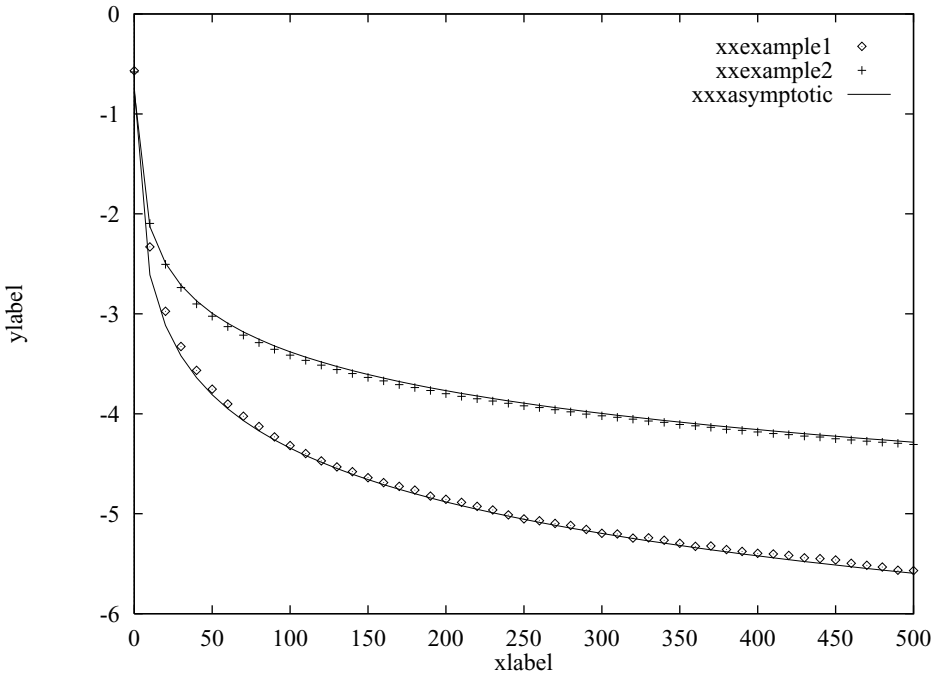
To illustrate how fast the asymptotic regime is reached we simulate the D-BMAP/D/1 queue with LRD arrival process characterized by  $\mathbf{P}\{\tau_A = j\} = (j+1)^{-s} - j^{-s}$ , and  $\phi(z) = 1 - \nu + \nu z$ . Here  $a = s - 1$ . For the first example  $s = 2.8$ ,  $\beta = 0.6$  and  $\nu = 0.3$ , and for the second example  $s = 2.3$ ,  $\beta = 0.8$  and  $\nu = 0.3$ . Hence the load of the system is about 0.68 for the first case and about 0.65 for the second case. As can be seen from Figure 1, the asymptotic regime is reached very quickly. The asymptotic behaviour of  $\mathbf{P}\{q = k\}$  is given by

$$\mathbf{P}\{q = k\} \sim \frac{a}{(s-1)} \frac{1}{\mathbf{E}[\tau_A] + \mathbf{E}[\tau_B]} \left( \frac{\phi'(1)^{s-1}}{1-\rho} + \phi'(1)^{s-2} \right) k^{1-s}.$$

### 3.2 The Tail Probabilities When $c > 1$

Contrary to the case  $c = 1$ , no closed form formula describing the asymptotic behaviour can be obtained when  $c > 1$ . The main reason is that an explicit formula for  $Q(z)$  like (11) does not exist for the D-BMAP/D/ $c$  with  $c > 1$ . Nevertheless it is possible to determine precisely the behaviour of the tail probabilities. It turns out that this behaviour depends on  $c$  and on the sizes of the LRD and the SRD shares of the traffic mix.

First we derive an expression for the generating function of the stationary buffer distribution  $q$ , denoted by  $Q(z)$ , from the Pollachek-Kinchin equation for the D-BMAP/D/ $c$  queue.



**Fig. 1.** Numerical example with  $c = 1$

**Calculating  $Q(z)$  When  $c > 1$ .** Recall that the Pollachek-Kinchin equation for the D-BMAP/D/ $c$  queue is given by

$$\mathbf{X}(z) = \sum_{l=0}^{c-1} \mathbf{x}_l \mathbf{D}(z) (z^c - z^l) (z^c \mathbf{I} - \mathbf{D}(z))^{-1}.$$

Whereas for  $c = 1$  the vector  $\mathbf{x}_0$  is known explicitly, here the vectors  $\mathbf{x}_0, \dots, \mathbf{x}_{c-1}$  can only be computed numerically, as is shown in [13, pg. 310–329]. This pre-

vents us from deriving a closed form formula for  $Q(z) = \mathbf{X}(z)\mathbf{e}$ . Define for  $l = 0, 1, \dots, c-1$ ,

$$\mathbf{w}^{(l)}(z) = \mathbf{x}_l \mathbf{D}(z) (z^2 \mathbf{I} - \mathbf{D}(z))^{-1},$$

with  $\mathbf{w}^{(l)}(z) = (w_0^{(l)} w_1^{(l)} \dots)$ . After some algebraic manipulations (see [3] for details) we obtain

$$\begin{aligned} \mathbf{w}^{(l)}(z)\mathbf{e} &= w_0^{(l)}(z) + w_1^{(l)}(z) \left[ 1 + \sum_{k=1}^{\infty} \left( \sum_{j>k} a_j \right) \left( \frac{\phi(z)}{z^{c-1}} \right)^k \right] \\ &\quad + \sum_{j=1}^{\infty} (\mathbf{x}_l)_j \sum_{k=j}^{\infty} \frac{1 - \sum_{i=1}^k a_i}{1 - \sum_{i=1}^{j-1} a_i} \left( \frac{\phi(z)}{z^{c-1}} \right)^{k-j+1}. \end{aligned} \quad (12)$$

The analysis of

$$Q(z) = \sum_{l=0}^{c-1} (z^c - z^l) \mathbf{w}^{(l)}(z) \mathbf{e}, \quad (13)$$

reveals two different types of asymptotic behaviour. If  $\phi'(1) > c-1$ , or equivalently, if the mean arrival rate of the background traffic exceeds  $c-1$ , then the tail probabilities decay according to a power-law, as is shown under Case 1. If the mean arrival rate of the background traffic is strictly less than  $c-1$ , then the tail probabilities decay approximately exponentially, as is demonstrated under Case 2. It turns out that the rate of this decay can be determined easily. For the case  $\phi'(1) = c-1$ , the transition point, no conclusions can be drawn.

**Case 1:  $\phi'(1) > c-1$ .** Since  $\phi'(1) > c-1$  we have

$$\left. \frac{d}{dz} \left( \frac{\phi(z)}{z^{c-1}} \right) \right|_{z=1} > 0.$$

Hence there exists an open set  $G \subset D(0, 1)$ , with  $D(0, 1)$  the closed complex unit disk, having the following properties:

$$\text{for each } z \in G: \left| \frac{\phi(z)}{z^{c-1}} \right| < 1; \quad (14)$$

$$\text{the interval } [\zeta, 1) \text{ belongs to } G \text{ for some } \zeta \in (0, 1). \quad (15)$$

Hence on  $G$  the representation (13) of  $Q(z)$  can be used. Since we focus on the influence of the LRD on-off source,  $\phi(z)$  is taken to be analytical on some open set containing  $D(0, 1)$ .

We obtain the asymptotic behaviour of the tail probabilities, as in 3.1, by examining the behaviour of  $Q'(z)$ . A detailed description of this analysis can be found in [3]. It is shown that for some  $C > 0$ ,

$$Q'(z) \sim \frac{C}{(1-z)^{3-s}} \text{ for } z \rightarrow 1-,$$

which implies

$$\mathbf{P}\{q > k\} \sim \frac{C(3-s)}{(s-2)\Gamma(4-s)} k^{2-s}.$$

The constant  $C$  depends on  $A(z)$ ,  $\beta$ ,  $\phi(z)$ ,  $c$  and the vectors  $\mathbf{x}_0, \dots, \mathbf{x}_{c-1}$ . Keep in mind that  $\mathbf{x}_0, \dots, \mathbf{x}_{c-1}$  need a numerical calculation.

**Case 2:  $\phi'(1) < c - 1$ .** Contrary to the case  $\phi'(1) > c - 1$ , there exists some  $z_0 > 1$  such that  $\phi(z_0) = z_0$ . We suppose that  $z_0 < \infty$ , as  $z_0 = \infty$  would imply that the background traffic generates at most  $c - 1$  arrivals in a slot. If this is the case no buffer is needed, because all arriving cells can be served instantaneously. For  $z \in (1, z_0)$ ,

$$\frac{\phi(z)}{z^{c-1}} < 1.$$

Since  $\phi(z)$  is a generating function, it follows that  $|\phi(z)| < |z^{c-1}|$ , for all complex number  $z$ , with  $1 < |z| < |z_0|$ . Furthermore, because  $\phi(z) \neq z^n$  for each  $n \geq 0$ , there exist at most a finite number of complex points  $z_i^*$  with modulus 1 such that  $|\phi(z_i^*)| = |z_i^*|$ . Let  $G$  be the maximal open set such that for  $z \in G$ ,

$$\left| \frac{\phi(z)}{z^{c-1}} \right| < 1.$$

Hence  $z \in G$ , for all  $1 < |z| < |z_0|$ . Furthermore  $G \cap D(0, 1)$  is non-empty. Notice that on  $G$  the function  $Q(z)$ , given by (13), is well-defined. Recall that  $q$  denotes the random variable associated with the stationary buffer distribution. Because  $G \cap D(0, 1)$  is non-empty,  $Q(z)$  is a representation of the  $z$ -transform  $\mathbf{E}[z^q]$  on  $G$ . Hence the power series  $\sum_{k=0}^{\infty} q_k z^k$ , associated with  $\mathbf{E}[z^q]$ , is holomorphic on the open disk with centre 0 and radius  $|z_0|$ . The point  $z_0$  represents a non-removable singularity. Since on  $G$ ,

$$\mathbf{E}[z^q] = \sum_{k=0}^{\infty} q_k z^k = Q(z),$$

it is possible to determine the asymptotic behaviour of  $q_k = \mathbf{P}\{q = k\}$  by examining the behaviour of  $Q'(z)$ . Define for  $y \in (0, 1)$ ,

$$f(y) = Q'(z_0 y).$$

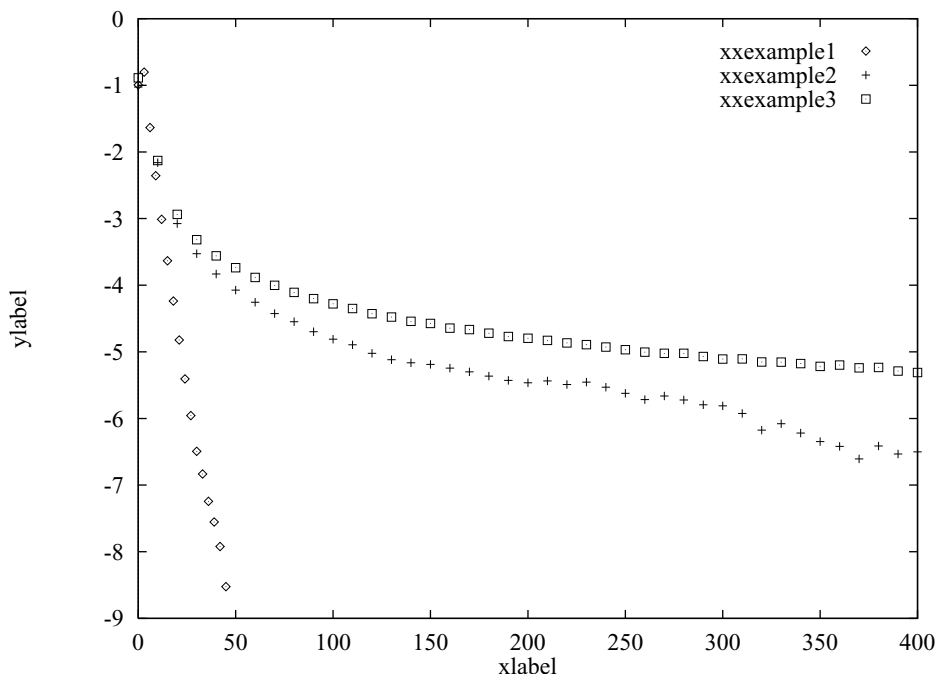
Analysing the behaviour of  $f(y)$  for  $y \rightarrow 1-$ , by using similar arguments as for the  $\phi'(1) > c - 1$  case, we obtain

$$q_1 + 2z_0 q_2 + \dots + n z_0^{n-1} q_n \sim C n^{3-s},$$

with  $C > 0$ . If the sequence  $(k z_0^{k-1} q_k)$  decreases, this implies

$$\mathbf{P}\{q = k\} \sim (3-s) C z_0^{-k} k^{1-s}.$$

In [4] this type of asymptotic behaviour is conjectured for similar queueing systems.



**Fig. 2.** Numerical example with  $c = 2$

**Numerical Example.** The tail probabilities for three examples, each one corresponding to a case considered above, are shown in Figure 2. The on-periods are distributed as  $\mathbf{P}\{\tau_A = j\} = (j+1)^{-s} - j^{-s}$ , the background traffic is generated by Poisson variables with parameter  $\lambda$ . For the first example  $s = 2.3$ ,  $\beta = 0.4$  and  $\lambda = 0.8$ , which results in a mean arrival rate of 1.5. Clearly this example is covered by Case 2. The second example has  $s = 2.6$ ,  $\beta = 0.4$  and  $\lambda = 1$ . Hence the mean arrival rate is 1.6. Since  $\lambda = 1$ , this example is a transition point case. In the third example  $s = 2.6$ ,  $\beta = 0.8$  and  $\lambda = 1.2$ , hence mean arrival rate is 1.5. This leads to a power law decay.

## 4 Conclusions

In this paper we have investigated the decay of the tail of the buffer occupancy distribution for a  $c$ -server queue which is fed by a mix of long range and short range dependent traffic. It turns out that the decay may be exponentially or according to a power law, depending on the composition of the traffic mix and the number of parallel servers  $c$ . The transition point occurs when the  $c > 1$  and the arrival rate of the SRD traffic is exactly  $c - 1$ . These results may be applied to traffic control schemes for LRD traffic in the following way. Consider SRD traffic mixed with LRD traffic which is spaced. The spacing distance is related



to the value of the number of parallel servers  $c$  in the model of the paper. The results give some insight under what conditions with respect to traffic mix and spacing distance, the tail of the buffer occupancy switches from a power law type to exponential.

## References

1. C. Blondia : A discrete-time batch markovian arrival process as B-ISDN traffic model. *Belgian Journal of Operations Research, Statistics and Computer Science*. **32** (1993) 3–23
2. T. Daniëls and C. Blondia. Asymptotic behavior of a discrete-time queue with long range dependent input. In *Proceedings of INFOCOM'99*. (1999)
3. T. Daniëls and C. Blondia. Tail Transitions in Queues with Long Range Dependent Input. Technical Report, [win-www.uia.ac.be/u/pats/publications.html](http://win-www.uia.ac.be/u/pats/publications.html). 2000
4. V. Dumas and A. Simonian. Asymptotic bounds for the fluid queue fed by sub-exponential on/off sources. Technical report, *Mathématiques Appliquées de Bordeaux*. (1998)
5. A. Erramilli, O. Narayan, and W. Willinger. Experimental queueing analysis with long-range dependent packet traffic. *IEEE/ACM Trans. on Networking*. **4** (1996) 209–223
6. W. Feller. *An Introduction to Probability and Its Applications, Volume II*. Wiley Series in Probability and Mathematical Statistics. John Wiley & Sons, Inc. (1966)
7. H. J. Fowler and W. E. Leland. Local area network traffic characteristics, with implications for broadband network congestion management. *IEEE JSAC* **9** (1991) 1139–1149
8. M. W. Garret and W. Willinger. Analysis, modeling and generation of self-similar VBR video traffic. In *Proceedings ACM Sigcomm'94* (1994) 269–280
9. D. Heath, S. Resnick, and G. Samorodnitsky. Heavy tails and long range dependence in on/off processes and associated fluid models. Technical report, Cornell University (1997)
10. P. R. Jelenković and A. A. Lazar. Asymptotic results for multiplexing on-off sources with subexponential on period. *Advances in Applied Probability* **31** (1999)
11. W. E. Leland, M. S. Taqqu, W. Willinger, and D. V. Wilson. On the self-similar nature of ethernet traffic (extended version). *IEEE/ACM Trans. on Networking*. **2** (1994) 1–15
12. B. M. Makarov, M. G. Goluzina, A. A. Lodkin, and A. N. Podkorytov. *Selected Problems in Real Analysis*. Translations of Mathematical Monographs, American Mathematical Society **107** (1992)
13. M. F. Neuts. *Structured stochastic matrices of the M/G/1-type and their applications*. Marcel Dekker (1989)
14. J. Roberts, U. Mocci, and J. Virtamo. *Broadband network teletraffic*. Springer (1996)
15. M. S. Taqqu, W. Willinger, and R. Sherman. Proof of a fundamental result in self-similar traffic modeling. *Computer Communication Review* **27** (1997) 5–23

# An Exact Algorithm for Calculating Blocking Probabilities in Multicast Networks

Eeva Nyberg, Jorma Virtamo, and Samuli Aalto

Laboratory of Telecommunications Technology  
Helsinki University of Technology

P.O.Box 3000, FIN-02015 HUT, Finland

{eeva.nyberg,jorma.virtamo,samuli.aalto}@hut.fi

**Abstract.** The paper deals with tree-structured point-to-multipoint networks, where users from infinite user populations at the leaf nodes subscribe to a variety of channels, offered by one source. The users joining the network form dynamic multicast connections that share the network resources. An exact algorithm for calculating end-to-end call blocking probabilities for dynamic connections is devised for this multicast model. The algorithm is based on the well-known algorithm for calculating blocking probabilities in hierarchical multiservice access networks, where link occupancy distributions are alternately convolved and truncated. The resource sharing of multicast connections requires the modification of the algorithm by using a new type of convolution, the OR-convolution. The exact algorithm for end-to-end call blocking probabilities enables us to study the accuracy of earlier results based on Reduced Load Approximation. The model is further extended to include background traffic, allowing the analysis of networks carrying mixed traffic e.g. multicast and unicast traffic.

## 1 Introduction

A multicast transmission originates at a source and, opposed to a unicast transmission, is replicated at various network nodes to form a tree-and-branch structure. The transmission reaches many different end-users without a separate transmission required for each user. A multicast connection has therefore a bandwidth saving nature. Blocking occurs in a network when, due to limited capacity, at least one link on the route is not able to admit a new call. Traditional mathematical models to calculate blocking probabilities in tree-structured networks exist for unicast traffic. Due to different resource usage, these models cannot directly be used for multicast networks where requests from different users arrive dynamically. Only recently, have mathematical models to calculate blocking probabilities in multicast networks been studied.

The past research has mainly been focused on blocking probabilities in multicast capable switches. Kim [6] studied blocking probabilities in a multirate multicast switch. Three stage switches were studied by Yang and Wang [11] and Listanti and Veltri [7]. Stasiak and Zwierzykowski [10] studied blocking in an

ATM node with multicast switching nodes carrying different multi-rate traffic (unicast and multicast), using Kaufman-Roberts recursion and Reduced Load Approximation. Admission control algorithms are studied in [9].

Chan and Geraniotis [1] have studied blocking due to finite capacity in network links. They formulated a closed form expression for time blocking probabilities in a network transmitting layered video signals. The model is a multipoint-to-multipoint model. The network consists of several video sources, where each source node can also act as a receiver. The video signals are coded into different layers defining the quality of the video signal received by the user. The traffic class is defined by the triplet: physical path, source node, and class of video quality. The behavior of each user is modeled as a two state Markov chain, with unique transition rates defined for each traffic class triplet.

Karvo et al. [3] and [4] studied blocking in a point-to-multipoint network with only one source node. The source is called the service center and it can offer a variety of channels, e.g. TV-channels. The users subscribing to the network may join and leave the channel at any time. The behavior of the user population defines the state probabilities at the links of the tree-structured network. The user population is assumed infinite and the requests to join the network arrive as from a Poisson process. The model studied in [3] considered the model in a simplified case where all but one link in a network have infinite capacity. They derived an exact algorithm to calculate both the channel and call blocking probability in this simplified case. Extending the model to the whole network was done only approximately in [4], where end-to-end blocking probabilities are estimated using the Reduced Load Approximation (RLA) approach.

This paper continues with section 2, where the single link case discussed in [3] and [4] is extended to a mathematical model for a multicast network with any number of finite capacity links. The section is divided into five parts. First, the notation is presented. Secondly, the model for a network with infinite link capacities is presented and thirdly, the OR-convolution used to convolve multicast state distributions in tree networks is introduced. Then, the main result is, an expression for the call blocking probability in a network with any number of finite capacity links is given and finally, the algorithm to calculate the call blocking probability is introduced. The algorithm is based on the well-known algorithm for calculating blocking probabilities in hierarchical multiservice access networks, where link occupancy distributions are alternately convolved and truncated. In section 3, comparisons between the exact solution and Reduced Load Approximation are carried through. The network model is extended to include non-multicast traffic as background traffic in section 4. The paper is concluded in section 5.

## 2 Network Model

### 2.1 Notation

The notation used throughout this paper is as follows. The set of all links is denoted by  $\mathcal{J}$ . Let  $\mathcal{U} \subset \mathcal{J}$  denote the set of leaf links. The leaf link and user

population behind the leaf link is denoted by  $u \in \mathcal{U} = \{1, \dots, U\}$ . The set of links on the route from leaf link  $u$  to the source is denoted by  $\mathcal{R}_u$ . The set of links downstream link  $j \in \mathcal{J}$  including link  $j$  is denoted by  $\mathcal{M}_j$ , while the set of downstream links terminating at link  $j \in \mathcal{J}$  is denoted by  $\mathcal{N}_j$ . The set of user populations downstream link  $j$  is denoted by  $\mathcal{U}_j$ . The set of channels offered by the source is  $\mathcal{I}$ , with channel  $i = 1, \dots, I$ . Let  $\mathbf{d} = \{d_i; i \in \mathcal{I}\}$ , where  $d_i$  is the capacity requirement of channel  $i$ . Here we assume that the capacity requirements depend only on the channel, but link dependencies could also be included into the model. The capacity of the link  $j$  is denoted by  $c_j$ . The different sets are shown in figure 1.

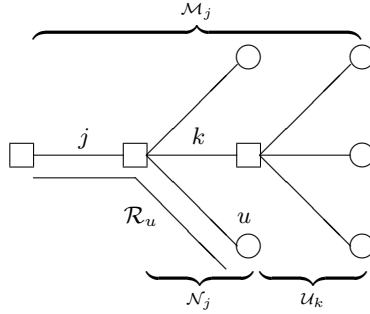


Fig. 1. An example network to show the notation used.

## 2.2 Network with Infinite Link Capacities

We first consider a network with all links having infinite capacity. Subscriptions to channel  $i$  behind leaf link  $u$  arrive from an infinite user population as from a Poisson process with intensity  $\lambda_{u,i} = \alpha_i \lambda_u$ , where  $\alpha_i$  is generated from a preference distribution for channel  $i$  and  $\lambda_u$  is the arrival intensity for user population  $u$ . The channel holding time is assumed to be generally distributed with mean  $1/\mu_i$ . In addition we denote the traffic intensity  $a_{u,i} = \alpha_i \lambda_u / \mu_i$ . Let the pair  $(u, i) \in \mathcal{U} \times \mathcal{I}$  denote a traffic class also called a connection. The connection state, which may be off or on, is denoted by  $X_{u,i} \in \{0, 1\}$ . It is shown in [3] that in a multicast network with all links having infinite capacity, the distribution of the number of users simultaneously connected to channel  $i$  is the distribution of the number of customers in an  $M/G/\infty$  queue. The state probability for a connection, is therefore

$$\pi_{u,i}(x_{u,i}) = P(X_{u,i} = x_{u,i}) = (p_{u,i})^{x_{u,i}} (1 - p_{u,i})^{1-x_{u,i}},$$

where  $p_{u,i} = 1 - e^{-a_{u,i}}$ .

In the infinite link capacity case, all connections are independent of each other. For leaf link  $u$ , the state probability has a product form and is

$$\pi_u(\mathbf{x}_u) = P(\mathbf{X}_u = \mathbf{x}_u) = \prod_{i \in \mathcal{I}} \pi_{u,i}(x_{u,i}),$$

where  $\mathbf{X}_u = (X_{u,i}; i \in \mathcal{I}) \in \mathcal{S}$  is the state vector for the leaf link, and  $\mathcal{S} = \{0, 1\}^I$  denotes the link state space.

The leaf link states jointly define the network state  $\mathbf{X}$ ,

$$\mathbf{X} = (\mathbf{X}_u; u \in \mathcal{U}) = (X_{u,i}; u \in \mathcal{U}, i \in \mathcal{I}) \in \Omega, \quad (1)$$

where  $\Omega = \{0, 1\}^{U \times I}$  denotes the network state space. For the whole network, the state probability is

$$\pi(\mathbf{x}) = P(\mathbf{X} = \mathbf{x}) = \prod_{u \in \mathcal{U}} \pi_u(\mathbf{x}_u),$$

as each user population is independent of each other.

**OR-convolution.** The leaf link state distributions jointly define the network state distribution, as was shown above. In order to calculate the link state distributions in a tree-structured network a convolution operation is needed. The resource sharing characteristic of multicast traffic requires a new type of convolution, the OR-convolution. Consider two downstream links  $s, t \in \mathcal{N}_v$  terminating at link  $v$ , where  $s, t, v \in \mathcal{J}$ . Channel  $i$  is idle in link  $v$  if it is idle in both links  $s$  and  $t$  and active in all other cases, which is equivalent to the binary OR-operation. In other words, for  $\mathbf{y}_s, \mathbf{y}_t \in \mathcal{S}$

$$\mathbf{y}_v = \mathbf{y}_s \oplus \mathbf{y}_t \in \mathcal{S}, \quad (2)$$

where the vector operator  $\oplus$  denotes the OR-operation taken componentwise. The OR-convolution, denoted by  $\otimes$ , is then the operation,

$$[f_s \otimes f_t](\mathbf{y}_v) = \sum_{\mathbf{y}_s \oplus \mathbf{y}_t = \mathbf{y}_v} f_s(\mathbf{y}_s) f_t(\mathbf{y}_t)$$

defined for any distributions  $f_s$  and  $f_t$ .

In a multicast link, the link state depends on the user states downstream the link. If a channel is idle in all links downstream link  $j$  it is off in link  $j$  and in all other cases the channel is active. The OR-operation on the network state gives the link state  $\mathbf{Y}_j = (Y_{j,i}; i \in \mathcal{I}) \in \mathcal{S}, j \in \mathcal{J}$  as a function of the network state,

$$\mathbf{Y}_j = \mathbf{g}_j(\mathbf{X}) = \bigoplus_{k \in \mathcal{U}_j} \mathbf{X}_k.$$

Similarly, the OR-convolution on the network state distribution gives the link state distribution. Thus, the state probability, denoted by  $\sigma_j(\mathbf{y}_j)$ , for  $\mathbf{y}_j \in \mathcal{S}$ , is equal to

$$\sigma_j(\mathbf{y}_j) = P(\mathbf{Y}_j = \mathbf{y}_j) = \left[ \bigotimes_{k \in \mathcal{U}_j} \pi_k \right](\mathbf{y}_j) = \begin{cases} \pi_j(\mathbf{y}_j) & , \text{ if } j \in \mathcal{U} \\ \left[ \bigotimes_{k \in \mathcal{N}_j} \sigma_k \right](\mathbf{y}_j) & , \text{ otherwise.} \end{cases}$$

When  $\mathbf{X} = \mathbf{x}$  the occupied capacity on the link  $j$  is  $\mathbf{d} \cdot \mathbf{g}_j(\mathbf{x})$ .

### 2.3 Blocking Probabilities in a Network with Finite Link Capacities

When the capacities of one or more links in the network are finite, the state spaces defined above are truncated according to the capacity restrictions. The network state  $\mathbf{X}$  defined in equation (1) is replaced by the truncated network state  $\tilde{\mathbf{X}} \in \tilde{\Omega}$ , where  $\tilde{\Omega}$  denotes the truncated state space

$$\tilde{\Omega} = \{\mathbf{x} \in \Omega \mid \mathbf{d} \cdot \mathbf{g}_j(\mathbf{x}) \leq C_j, \forall j \in \mathcal{J}\}.$$

The insensitivity [5] and truncation principles [2] apply for this product form network, and for the truncated system the state probabilities of the network differ only by the normalization constant  $G(\tilde{\Omega}) = \sum_{\mathbf{x} \in \tilde{\Omega}} \pi(\mathbf{x})$ . The state probabilities of the truncated system are therefore

$$\tilde{\pi}(\mathbf{x}) = P(\tilde{\mathbf{X}} = \mathbf{x}) = \frac{\pi(\mathbf{x})}{G(\tilde{\Omega})}, \text{ for } \mathbf{x} \in \tilde{\Omega}.$$

When the capacity on the links is finite, blocking occurs. Due to Poisson arrivals, the call blocking probability is equal to the time blocking probability of the system. A call in traffic class  $(u, i)$  is blocked if there is not enough capacity in the network to set up the connection. Note that, once the channel is active on all links belonging to the route  $R_u$  of user population  $u$ , no extra capacity is required for a new connection. Let us define another truncated set  $\tilde{\Omega}_{u,i} \subset \tilde{\Omega}$  with a tighter capacity restriction for links with channel  $i$  idle,

$$\tilde{\Omega}_{u,i} = \{\mathbf{x} \in \Omega \mid \mathbf{d} \cdot (\mathbf{g}_j(\mathbf{x}) \oplus (\mathbf{e}_i 1_{j \in \mathcal{R}_u})) \leq C_j, \forall j \in \mathcal{J}\},$$

where  $\mathbf{e}_i$  is the  $I$ -dimensional vector consisting of only zeroes except for a one in the  $i$ th component and  $1_{j \in \mathcal{R}_u}$  is the indicator function equal to one for  $j \in \mathcal{R}_u$  and zero otherwise. This set defines the states where blocking does not occur when user  $u$  requests a connection to channel  $i$ . The call blocking probability  $b_i^c$  for traffic class  $(u, i)$  is thus,

$$b_{u,i}^c = 1 - P(\tilde{\mathbf{X}} \in \tilde{\Omega}_{u,i}) = 1 - \frac{G(\tilde{\Omega}_{u,i})}{G(\tilde{\Omega})}. \quad (3)$$

This approach requires calculating two sets of state probabilities: the set of non-blocking states appearing in the numerator and the set of allowed states appearing in the denominator of equation (3).

A multicast network is a tree-type network, and much of the theory in calculating blocking probabilities in hierarchical multiservice access networks [8] can be used to formulate the end-to-end call blocking probability in a multicast network as well.

### 2.4 The Algorithm

As in the case of access networks, the blocking probability can be calculated by recursively convolving the state probabilities of individual links from the

leaf links to the origin link. At each step, the state probabilities are truncated according to the capacity restriction of the link.

In order to calculate the denominator of equation (3), let us define a new subset  $\tilde{\mathcal{S}}_j$  of set  $\mathcal{S}$ ,

$$\tilde{\mathcal{S}}_j = \{\mathbf{y} \in \mathcal{S} \mid \mathbf{d} \cdot \mathbf{y} \leq C_j\}, \text{ for } j \in \mathcal{J}.$$

The corresponding truncation operator acting on any distribution  $f$  is

$$T_j f(\mathbf{y}) = \quad (4)$$

Let

$$Q_j(\mathbf{y}_j) = P(\mathbf{Y}_j = \mathbf{y}_j; \mathbf{Y}_k \in \tilde{\mathcal{S}}_k, \forall k \in \mathcal{M}_j), \text{ for } \mathbf{y}_j \in \mathcal{S}. \quad (5)$$

It follows that the  $Q_j(\mathbf{y})$ 's can be calculated recursively,

$$Q_j(\mathbf{y}) = \begin{cases} T_j \pi_j(\mathbf{y}) & , \text{ if } j \in \mathcal{U} \\ T_j [\bigotimes_{k \in \mathcal{N}_j} Q_k](\mathbf{y}) & , \text{ otherwise.} \end{cases}$$

Note that, if the capacity constraint of link  $j \in \mathcal{M}_j$  is relaxed, then the branches terminating at link  $j$  are independent, and the jointly requested channel state can be obtained by the OR-convolution. The effect of the finite capacity  $C_j$  of link  $j$  is then just the truncation of the distribution to the states for which the requested capacity is no more than  $C_j$ .

The state sum  $G(\tilde{\Omega})$  needed to calculate the blocking probability in equation (3) is equal to

$$G(\tilde{\Omega}) = \sum_{\mathbf{y} \in \mathcal{S}} Q_J(\mathbf{y}),$$

where  $Q_J$  is the probability (5) related to the common link  $j = J$ .

Similarly for the numerator of equation (3), let  $\tilde{\mathcal{S}}_j^{u,i} \subset \tilde{\mathcal{S}}_j$  be defined as the set of states on link  $j$  that do not prevent user  $u$  from connecting to multicast channel  $i$ . In other words

$$\tilde{\mathcal{S}}_j^{u,i} = \{\mathbf{y} \in \mathcal{S} \mid \mathbf{d} \cdot (\mathbf{y} \oplus (\mathbf{e}_i 1_{j \in \mathcal{R}_u})) \leq C_j\}, \text{ for } j \in \mathcal{J}.$$

The truncation operator is then

$$T_j^{u,i} f(\mathbf{y}) = f(\mathbf{y}) 1_{\mathbf{y} \in \tilde{\mathcal{S}}_j^{u,i}} \quad (6)$$

The non-blocking probability of link  $j$  is

$$Q_j^{u,i}(\mathbf{y}_j) = P(\mathbf{Y}_j = \mathbf{y}_j; \mathbf{Y}_k \in \tilde{\mathcal{S}}_k^{u,i}, \forall k \in \mathcal{M}_j), \text{ for } \mathbf{y}_j \in \mathcal{S}. \quad (7)$$

Similarly, as above, it follows that

$$Q_j^{u,i}(\mathbf{y}) = \begin{cases} T_j^{u,i} \pi_j(\mathbf{y}) & , \text{ if } j \in \mathcal{U} \\ T_j^{u,i} [\bigotimes_{k \in \mathcal{N}_j} Q_k^{u,i}](\mathbf{y}) & , \text{ otherwise.} \end{cases}$$

The state sum in the numerator of equation (3) is then

$$G(\tilde{\Omega}_{u,i}) = \sum_{\mathbf{y} \in \mathcal{S}} Q_J^{u,i}(\mathbf{y}),$$

where  $Q_J^{u,i}$  is the probability (7) related to the common link  $j = J$ .

The blocking probability in equation (3) is therefore

$$b_{u,i}^c = 1 - \frac{\sum_{\mathbf{y} \in \mathcal{S}} Q_J^{u,i}(\mathbf{y})}{\sum_{\mathbf{y} \in \mathcal{S}} Q_J(\mathbf{y})}. \quad (8)$$

The complexity of the algorithm increases exponentially with the number of channels, as the number of states in the distributions to be convolved is  $2^I$ . Therefore the use of RLA as a computationally simpler method is studied.

**Single Finite Capacity Link.** The single link model by Karvo et al. [3] is a special case of the network model presented. In a network, with all but one link with infinite capacity, and thus only one user population  $u$  that experiences blocking ( $b_{u,i}^c = b_i^c$ ), it follows that equation (8) transforms into equation (17) in [3].

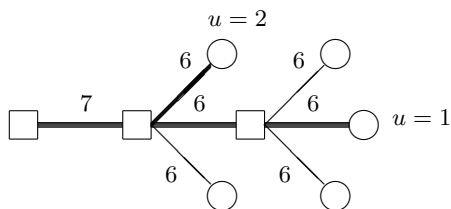
$$\begin{aligned} b_i^c &= \frac{\sum_{j=C-d_i+1}^C \pi_j^{(x_i=0)}}{\sum_{j=0}^C \pi_j} = \frac{(1-p_i) \sum_{j=C-d_i+1}^C \pi_j^{(i)}}{(1-p_i) \sum_{j=0}^C \pi_j^{(i)} + p_i \sum_{j=0}^{C-d_i} \pi_j^{(i)}} \\ &= \frac{(1-p_i) \sum_{j=C-d_i+1}^C \pi_j^{(i)}}{(1-p_i) \sum_{j=0}^C \pi_j^{(i)} + p_i (\sum_{j=0}^C \pi_j^{(i)} - \sum_{j=C-d_i+1}^C \pi_j^{(i)})} \\ &= \frac{(1-p_i) B_i^c}{1-p_i + p_i(1-B_i^c)} = \frac{B_i^c}{(1-B_i^c)(e^{a_i} - 1) + 1}, \end{aligned}$$

where  $\pi_j$  is the link occupancy distribution for an infinite system,  $\pi_j^{(x_i=0)}$  is the link occupancy distribution restricted to the states with channel  $i$  off, and  $\pi_j^{(i)}$  is the link occupancy distribution of a system with channel  $i$  removed using the same notation as in [3].

### 3 Comparisons Between the Exact Model and RLA

The calculation of end-to-end call blocking probabilities for multicast networks was done approximately using the RLA-algorithm in [4], where the details of this well-known algorithm in the case of multicast networks are given. The exact algorithm derived in the previous section allows us to study the accuracy of RLA. To this end, we consider the example network depicted in figure 2. Due to symmetry, the five different user populations reduce to two distinctly different





**Fig. 2.** Network used to compare exact results with results given by the RLA-algorithm.

user populations and hence routes. The capacities of each link are also depicted in the figure. The links are numbered in the following way. The leaf link of user one has  $u, j = 1$ , the leaf link of user two has  $u, j = 2$ , the middle link has  $j = 3$ , and the common link has  $j = 4$ . Comparisons were made between the exact solution and the RLA-algorithm. The number of channels offered is eight. Each channel requires one unit of capacity. The common link in the network has a capacity of seven units. All other links have a capacity of six units. The end-to-end call blocking probabilities are calculated for the least used channel using a truncated geometric distribution for the channel preference

$$\alpha_i = \frac{p(1-p)^{i-1}}{1-(1-p)^I},$$

with parameter  $p = 0.2$ . The mean holding time is the same for all channels,  $1/\mu = 1$ . In addition, the arrival intensity is the same for both user populations,  $\lambda_u = \lambda$  and consequently, the traffic intensity  $a = \lambda/\mu$  is the same for both user populations.

The results are given in table 1. The comparison was also done for multiservice traffic, where the capacity requirement is one for odd channels and two for even channel numbers. The capacity of the common link was eleven units and those of the other links were nine units. The results are given in table 2.

The results confirm the comparisons made in [4]. RLA-algorithm yields blocking probabilities of the same magnitude as the exact method. As a rule, RLA gives larger blocking values for both routes. For route 2, RLA gives good results.

**Table 1.** Call blocking probabilities for the network in figure 2.

	Route1 ( $u = 1$ )			Route2 ( $u = 2$ )		
$a$	Exact	RLA	error	Exact	RLA	error
1.0	0.0056	0.0064	14 %	0.0027	0.0028	4 %
1.1	0.0084	0.0098	17 %	0.0041	0.0044	7 %
1.2	0.0121	0.0141	17 %	0.0060	0.0064	8 %
1.3	0.0166	0.0195	17 %	0.0083	0.0090	8 %
1.4	0.0220	0.0260	18 %	0.0112	0.0121	8 %
1.5	0.0282	0.0336	19 %	0.0146	0.0157	8 %
2.0	0.0715	0.0868	21 %	0.0382	0.0416	9 %

**Table 2.** Call blocking probabilities for the network in figure 2, with capacity requirements  $c_{odd} = 1$  and  $c_{even} = 2$ .

		Route1 ( $u = 1$ )			Route2 ( $u = 2$ )		
Channel	$a$	Exact	RLA	error	Exact	RLA	error
7	1.0	0.0051	0.0058	14 %	0.0019	0.0022	16 %
8	1.0	0.0127	0.0139	9 %	0.0028	0.0029	4 %
7	1.3	0.0138	0.0162	17 %	0.0058	0.0068	17 %
8	1.3	0.0318	0.0355	12 %	0.0086	0.0092	7 %
7	1.5	0.0226	0.0268	19 %	0.0100	0.0118	18 %
8	1.5	0.0499	0.0566	13 %	0.0151	0.0162	7 %
7	2.0	0.0536	0.0645	20 %	0.0255	0.0299	17 %
8	2.0	0.1101	0.1276	16 %	0.0400	0.0431	8 %

This is because the route is very short, and the assumption of independence between the links is not violated severely.

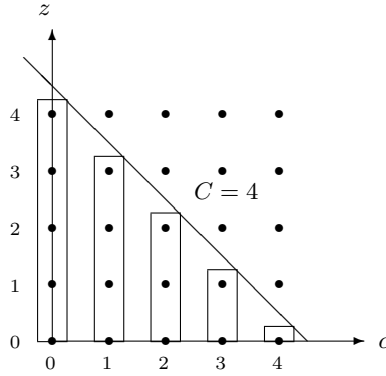
## 4 Including Background Traffic

The networks considered until now were assumed to transfer only multicast traffic. The model can, however, be extended to cover networks with mixed traffic. In this case, the network transfers, in addition to multicast traffic, non-multicast traffic that is assumed independent on each link. The distribution does not depend on the multicast traffic in the link and the traffic in the other links. The non-multicast traffic in link  $j$  is assumed to be Poisson with a traffic intensity  $A_j$ . The capacity requirement is equal to one unit of capacity. The link occupancy distribution of the non-multicast traffic in a link with infinite capacity is thus,

$$q_j(z) = \frac{(A_j)^z}{z!} e^{-A_j}.$$

The inclusion of non-multicast traffic affects only the truncation step of the algorithm presented in section 2.4. The state probabilities are defined as in section 2. The state probabilities of the link states that require more capacity than available on the link are set to zero as before. However, the state probabilities of the states that satisfy the capacity restriction of the link are altered, as the available capacity on the link depends on the amount of non-multicast traffic on the link. Another way of describing the relationship between the two different types of traffic, is to consider them as two traffic classes in a two dimensional link occupancy state space as is shown in figure 3. The traffic classes are independent of each other. The capacity of the link is the linear constraint of this state space.

We notice that the marginal distribution of the capacity occupancy of the multicast traffic is weighted by the sums over the columns of the occupancy probabilities of the background traffic. If the multicast traffic occupies  $c = \mathbf{d} \cdot \mathbf{y}_j$  units of capacity, and the link capacity is  $C_j$ , then possible non-multicast traffic



**Fig. 3.** Shaping of the marginal distribution of the capacity occupancy when background traffic is included in the model.

states on the link are those with  $z \leq C_j - c$ , where  $z$  is the number of non-multicast calls. Therefore, the truncation functions presented in equations (4) and (6) must be replaced by the operators

$$\hat{T}_j f(\mathbf{y}) = \sum_{z=0}^{C_j - \mathbf{d} \cdot \mathbf{y}} q_j(z) f(\mathbf{y}), \quad \text{and} \quad \hat{T}_j^{u,i} f(\mathbf{y}) = \sum_{z=0}^{C_j - \mathbf{d} \cdot (\mathbf{y} \oplus (\mathbf{e}_i 1_{j \in \mathcal{R}_u}))} q_j(z) f(\mathbf{y})$$

The algorithm differs therefore only by the truncation function used,

$$\hat{Q}_j(\mathbf{y}) = \begin{cases} \hat{T}_j \pi_j(\mathbf{y}) & , \text{ if } j \in \mathcal{U} \\ \hat{T}_j [\bigotimes_{k \in \mathcal{N}_j} \hat{Q}_k](\mathbf{y}) & , \text{ otherwise.} \end{cases}$$

$$\hat{Q}_j^{u,i}(\mathbf{y}) = \begin{cases} \hat{T}_j^{u,i} \pi_j(\mathbf{y}) & , \text{ if } j \in \mathcal{U} \\ \hat{T}_j^{u,i} [\bigotimes_{k \in \mathcal{N}_j} \hat{Q}_k^{u,i}](\mathbf{y}) & , \text{ otherwise.} \end{cases}$$

The call blocking probability in equation (3) is again obtained by two series of convolutions and truncations from the leaf links to the common link  $J$ . The end-to-end call blocking probability of the network is

$$\hat{b}_{u,i}^c = 1 - \frac{\sum_{\mathbf{y} \in \mathcal{S}} \hat{Q}_J^{u,i}(\mathbf{y})}{\sum_{\mathbf{y} \in \mathcal{S}} \hat{Q}_J(\mathbf{y})}.$$

#### 4.1 Numerical Results

The end-to-end call blocking probability was calculated using the same network as in section 3, figure 2. The intensity of the non-multicast traffic was set to  $A_j = 0.1$  for all links. Table 3 shows the end-to-end call blocking probability for a network with only multicast traffic and for a network transferring multicast and

non-multicast traffic. Table 4 shows the end-to-end call blocking probabilities when the multicast traffic requires double the capacity compared to the non-multicast traffic.

The intensity of non-multicast traffic stays the same, as the intensity of the multicast traffic increases. Clearly, the blocking probabilities are affected less, as the intensity of the multicast traffic increases. This can also be seen by studying the relative change in blocking probabilities shown in tables 3 and 4. The effect of the non-multicast traffic to the blocking probability is of the same magnitude on both routes. From table 3 we see that an inclusion of unicast traffic with one tenth the intensity  $a = 1.0$  of the multicast traffic almost doubles the blocking probability. From table 1 the blocking probability increases by a factor of 1.5, when the traffic intensity  $a$  is increased from 1.0 to 1.1. These two cases are not equivalent as the background traffic is assumed independent of the multicast traffic, but give a good reference to the effect background traffic has on end-to-end blocking probabilities.

**Table 3.** End-to-end blocking probabilities for the network in figure 2 with background traffic and multicast traffic.

	Route1 ( $u = 1$ )			Route2 ( $u = 2$ )		
a	Multicast	Background	Rel. change	Multicast	Background	Rel. change
1.0	0.0056	0.0109	1.95	0.0027	0.0053	1.96
1.2	0.0121	0.0206	1.70	0.0060	0.0105	1.75
1.4	0.0220	0.0341	1.55	0.0112	0.0177	1.58
2.0	0.0715	0.0927	1.30	0.0382	0.0501	1.31

**Table 4.** End-to-end blocking probabilities for the network in figure 2 with background traffic requiring one unit and multicast traffic requiring two units of capacity.

	Route1 ( $u = 1$ )			Route2 ( $u = 2$ )		
a	Multicast	Background	Rel. change	Multicast	Background	Rel. change
1.0	0.0056	0.01	1.79	0.0027	0.0049	1.81
1.2	0.0121	0.0195	1.61	0.0060	0.0099	1.65
1.4	0.0220	0.0328	1.49	0.0112	0.0171	1.53
2.0	0.0715	0.0914	1.28	0.0382	0.0495	1.30

## 5 Conclusions

The paper presented a new algorithm for exactly calculating end-to-end blocking probabilities in tree-structured multicast networks. The algorithm is based on the well-known algorithm for calculating blocking probabilities in hierarchical multiservice access networks. The multicast traffic characteristics were taken into account in the convolution step of the algorithm, using the new OR-convolution.

Calculating the exact solution for the end-to-end call blocking probability, however, becomes infeasible as the number of channels increases. In contrast to ordinary access networks, the aggregate one dimensional link occupancy description is not sufficient, since in the multicast network it is essential to do all calculations in the link state space, with  $2^I$  states. This is due to the resource sharing property of multicast traffic, namely the capacity in use on a link increases only if a channel is requested when the channel is idle. The use of RLA was studied, as the complexity of the RLA-algorithm does not depend critically on the number of channels in the network. RLA method used in [4], however, gives larger blocking probabilities. Even for small networks, the errors are around 15 %. The network model and the algorithm for calculating call blocking probabilities were further broadened to include background traffic in addition to multicast traffic.

We leave for further research the study of new approximation methods for calculating end-to-end call blocking probabilities. Fast implementation of the exact algorithm presented should also be investigated. At present, the model also assumes an infinite user population behind each leaf link. The model can be generalized to allow a finite user population behind a leaf link and is a subject for further study.

## References

1. Chan W.C., Geraniotis E., Tradeoff Between Blocking and Dropping in Multicasting Networks, IEEE International Conference on Communications, Conference Record, Converging Technologies for Tomorrow's Applications pp. 1030-1034, 1996.
2. Hui J., "Switching and traffic theory for integrated broadband networks", Kluwer Academic Publishers, Boston, 1990.
3. Karvo J., Virtamo J., Aalto S., Martikainen O., Blocking of dynamic multicast connections in a single link, Proc. of Broadband Communications '98, ed. P. Kühn, R. Ulrich, IFIP, pp. 473-483, 1998.
4. Karvo J., Virtamo J., Aalto S., Martikainen O., Blocking of Dynamic Multicast Connections, In 4th INFORMS Telecommunications Conference, Boca Raton, Florida, March 1998. To appear in Telecommunication Systems.
5. Kelly F.P., "Reversibility and Stochastic Networks", John Wiley & Sons, 1979.
6. Kim C.K., Blocking Probability of Heterogeneous Traffic in a Multirate Multicast Switch, IEEE Journal on Selected Areas in Communications 14(2), 1996, pp. 374-385.
7. Listanti M., Veltri L., Blocking Probability of Three-Stage Multicast Switches, IEEE International Conference on Communications, Conference Record, 1998, pp. 623-629.
8. Ross K. W., "Multiservice Loss Models for Broadband Telecommunication Networks", Springer-Verlag, London, 1995.
9. Shacham N., Yokota H., Admission Control Algorithms for Multicast Sessions with Multiple Streams IEEE Journal on Selected Areas in Communications 15(3), 1997, pp. 557-566.
10. Stasiak M. and Zwierzykowski P., Analytical Model of ATM Node with Multicast Switching, Mediterranean Electrotechnical Conference, 1998, pp. 683-687.
11. Yang Y., On Blocking Probability of Multicast Networks, IEEE Transactions on Communications 46 (7), pp. 957-968, July 1998.

# Approximations for Call-Blocking Probabilities in Multirouting Multicasting Multirate Loss Networks

Tibor Cinkler and László Ast

High-Speed Networks Laboratory  
Department of Telecommunications and Telematics  
Budapest University of Technology and Economics  
Pázmány Péter sétány 1/D, H-1117 Budapest, Hungary  
`cinkler@ttt-atm.ttt.bme.hu`

**Abstract.** For the end-to-end call blocking probability computation in multi-rate loss networks the so called reduced load approximation under link independence assumption is often used, because it allows derivation of analytical results. However, its accuracy and extendibility to multirouting or multicasting networks (like the B-ISDN) is seldom studied. This paper proposes new analytical methods for approximating the call blocking probabilities for alternative routing and point-to-multipoint connections. The results are derived for different cases and techniques and further generalised for Whitt's summation bound.

## 1 Introduction

The exact (or good approximate) and fast calculation of end-to-end blocking probabilities in modern telecommunication networks is important, because it is the primary measure of the grade of service the network provides to its users [1,2,3,4]. In broadband integrated services digital networks (B-ISDN), where calls of different service classes are routed between two or more different points, the traditional methods, elaborated for single-rate circuit switched (telephone) networks are no longer adequate. The most frequently used originator-destination (OD) pair blocking evaluation technique is the reduced load (RL) approximation. It assumes link independence: the links of a multihop route block independently from each other, i.e., the events "link  $i$  blocks" and "link  $j$  blocks" are independent of each other, even if both links are of the same route. This assumption is luring, because it allows the calculation of route blocking probabilities by means of product forms [5,6]. The reduced load approximation determines the route blocking probabilities using blocking probabilities of those links which are along that route [7].

The validity of the link independence assumption has been studied by Roberts, Dziong, Pióro, Blaabjerg and others, and a number of valuable results exist [7,8,9]. The RL approximation has been generalised for the multi-rate case where the traffic sources generate the load at different (bps) rates [7]. Unfortunately,

very few results consider its generalisation to multiroute cases (where a set of routes is offered to each OD pair). Furthermore, to our knowledge it is not discussed how they can be applied to multicast routing - an important routing method, where each connection can be built up between more than two or three users and is expected to be in wide use in future multiservice networks. The obtained results were published in [10,11].

In Section 2 we present analytical methods for determining link and path blocking probabilities for multi-rate case. Then we generalise these results for tree blocking probabilities in case of multicasting, and for OD-pair blocking probabilities in case of multirouting (Alternative Routing (AR) only).

## 2 Blocking Probability Evaluation Methods

A route  $r$  is a subset of the set of those logical links that belong to the same logical network. In principle, it can be taken as an arbitrary subset but here they will be simple paths connecting OD (Origin-Destination) pairs and trees connecting a subset of all nodes. By convention, each route carries only calls of a single traffic class  $m$ . That is, if several traffic classes are to be carried, they are represented by parallel routes. The incidence of routes, links and traffic types is expressed by the indicator variables

$$A_{mjr} = \begin{cases} 1 & \text{when route } r \text{ uses link } j \text{ and carries traffic type } m \\ 0 & \text{otherwise} \end{cases} \quad (1)$$

$A_{mjr}$  is *not* to be interpreted as the amount of bandwidth that route  $r$  requires on link  $j$ . For that purpose we use other variables:  $d_{mj}$  will denote the amount of bandwidth (capacity) that a call belonging to traffic type  $m$  requires on link  $j$ . By this notation we implicitly assume that all routes that carry traffic of type  $m$  require the same amount of bandwidth on link  $j$ . Since the bandwidth requirement is associated with the traffic type, this is not seen as a restriction. On the other hand, we allow the bandwidth requirement of calls on a given route to vary along the links of the route. In fact, this is needed if the concept of effective or equivalent bandwidth (see e.g., [12]) is adopted and the involved links have different capacities.

Let  $\nu_r$  be the offered traffic to route  $r$ . Let  $\nu_{(v,p,m)}$  be the aggregated offered traffic of class  $m$  to node pair  $p$  in virtual network  $v$ . Then the traffic carried over route  $r$  can be expressed as

$$\lambda_r = \nu_r(1 - L_r) \quad (2)$$

where  $L_r$  is the end-to-end route blocking probability for traffic on route  $r$ . Clearly, this route blocking probability is defined as the probability of the event that at least one link is blocked along the route.

For traditional single-rate networks it is known that alternative routing (AR) offers some advantages compared to either simple load sharing or fixed routing. The model presented here can quite easily be extended to either sequential or

dynamic alternative routing, at least if the overflow traffic is approximated by Poisson traffic. This approximation is justified in the case where only a minor amount of traffic overflows. Therefore the approximation for Dynamic AR (DAR) will be more accurate in general than for Sequential AR (SAR), since DAR decreases the overflow by dynamically changing the primary route.

The call blocking probabilities can be considered at different levels:

- link blocking,
- route blocking,
- tree blocking, and
- OD pair (end-to-end) blocking

The call blocking probabilities  $L_r$  can for a loss network operating under fixed routing receiving Bernoulli-Poisson-Pascal traffic, in principle be found since such a network obeys a product form solution [13]. However, the numerical complexity is prohibitively high for networks of reasonable sizes, and in practice approximations need to be applied.

Here we shall apply two related techniques which both give the route blocking probabilities whenever the blocking probabilities on individual links are known. Furthermore, we approximate OD pair blocking probabilities for multirouting techniques based on the blocking probabilities of routes of the considered OD pair.

### 3 Link Blockings

The analytical blocking probability evaluation for routes, trees and OD-pairs are based on evaluation of link blocking probabilities. Here we assume Complete Sharing, where the traffic of different connections belonging to different traffic classes share the common resources. These methods can simply be generalised for case of Complete Partitioning, Trunk Reservation and other resource sharing/partitioning techniques.

The natural choice for evaluating link-blocking probabilities in a multi-rate network would have been to apply the multi-dimensional Erlang formula, which is very accurate. However, increasing the number of different traffic classes the state-space grows exponentially. Therefore, the projection of this multidimensional state-space to a one-dimensional state-space is preferred.

Consider link  $j$  of capacity  $C_j$  in isolation receiving traffic from the routes going through link  $j$ . With  $\nu_{m,j}$  being the total call arrival rate from traffic class  $m$  on link  $j$ , with  $d_{m,j}$  being the bandwidth demand of calls from class  $m$ , if the holding time as before is assumed equal to 1 for all traffic classes, then the occupancy distribution of link  $j$  is found from the Kaufman-Roberts recursion formula (also called stochastic Knapsack) which states (see, e.g., [14])

$$np_{j,n} = \sum_m \nu_{m,j} d_{m,j} p_{j,n-d_{m,j}} \quad \text{and} \quad \sum_{n=0}^{C_j} p_{j,n} = 1 \quad (3)$$



where  $p_{j,n}$  is the probability of  $n$  occupied bandwidth units (circuits) on link  $j$ . The blocking probability for traffic class  $m$  can now be found as

$$B_{mj} = \sum_{n=C_j-d_m+1}^{C_j} p_{j,n} \quad (4)$$

For optimisation of large networks, this formula is too complex. The complexity originates from two facts. First the underlying Markov process is multi-dimensional, and there exists no simple relationship between the blocking probabilities of the individual traffic classes enabling a computation of all the blocking probabilities by just computing the blocking of a single traffic class (e.g., the most narrow one). Secondly, it is a recursive formula and this implies that the complexity increases with increasing capacities of the links.

As a somewhat simpler but in many cases still quite accurate formula, the Pascal approximation can be used. Here, the mean and variance of the occupied bandwidth in case of infinite capacity links is found and a negative binomial (Pascal) distribution with same mean and variance is used, as an approximation of the occupancy distribution. By the usual truncation and by applying (4) with  $p_{j,n}$  found from the Pascal approximation the individual blocking probabilities can be found. See, e.g., [15] and [16] for further details. Even though the Pascal approximation is simpler it is also burdened with increasing complexity in increasing capacities.

For high capacity networks where both these formulae become too complicated a simple Gaussian approximation as used and explained in [17] can be useful.

Finally, based on the behavior of the blocking probabilities in the limiting regime [18] where offered traffic and link capacities tend to infinity by the same speed, a very simple approximation could be

$$(1 - B_{m,j}) \approx (1 - B_{1,j})^{(d_{m,j}/d_{1,j})} \quad (5)$$

i.e., a call requiring a factor of  $d$  more bandwidth units than the most narrow call has a blocking probability as if it was set up as  $d$  sequential narrow band calls. For the case where the fraction between the biggest and smallest bandwidth demand is not far from 1 this approximation can be applicable. However as this fraction increases the accuracy deteriorates.

## 4 Route Blockings

### 4.1 Reduced Load and Link Independence

The *first method* to be presented is the classical *reduced load and link independence* assumption, see, e.g., [18] and [16]. Here blocking of individual links is assumed to take place independently of other link blockings and the load offered to link  $j$  consists not of the total load offered but only the traffic not blocked at other links. To be more precise let  $B_{m,j}$  denote the blocking probability of

traffic class  $m$  on link  $j$ , and let  $E_{m,j}(\rho_1, \dots, \rho_M; C_j)$  be the blocking function which gives the blocking probability of traffic class  $m$  on link  $j$  when the offered class  $m$  bandwidth demand on link  $j$  is  $\rho_m$  and link capacity is  $C_j$ . Then

$$B_{mj} = E_{mj}(\rho_{1j}, \dots, \rho_{Mj}; C_j) \text{ for all } m \text{ and } j \quad (6)$$

$$\lambda_r = \nu_r(1 - L_r) \text{ where } (1 - L_r) = \prod_{m,k} (1 - B_{mk})^{A_{mkr}} \quad (7)$$

$$\rho_{mj} = d_{mj} \sum_r (1 - B_{mj})^{-1} \lambda_r A_{mjr} = d_{mj} \nu_{m,j} \text{ for all } m \text{ and } j \quad (8)$$

where  $L_r$  is the end-to-end blocking probability for traffic on route  $r$ ,  $\lambda_r$  is the carried traffic on route  $r$  and  $\nu_{m,j}$  is the offered class  $m$  traffic to link  $j$  under the reduced load approximation.

Clearly, this route blocking probability is defined as the probability of the event that at least one link is blocked along the route.

As seen from (7) then the route blocking probability on any route is easily derived when the link blocking probabilities  $B_{mk}$  are known, and they are given as a solution to (6), (7) and (8). By Brouwer's fixed point theorem a solution is known to exist but unfortunately, in the multi-rate case multiple solutions cannot in general be ruled out.

## 4.2 Whitt's Summation Bound

The idea to the *second method* originates from a result valid for single slot circuit switched networks proved by Whitt in [19].

The result of Whitt (Theorem 1 and Corollary 1.2 in [19]) states that in a single rate loss network operating under fixed routing and receiving Poissonian traffic, the exact route blocking probability  $L_r^{exact}$  is upper bounded by the sum of the link blocking probabilities on the links  $j$  which route  $r$  passes, i.e.,

$$L_r^{exact} \leq \sum_{j \in r} \tilde{B}_j \quad (9)$$

where  $\tilde{B}_j$  is the (single class) blocking probability on link  $j$  under the assumption that all traffic offered to the routes passing through link  $j$  is also offered to link  $j$ , i.e., *not* blocked at other links, i.e., the traffic offered to link  $j$  is  $\sum_r \nu_r A_{jr}$  (the traffic class index  $m$  is omitted).

The main advantage of the generalised Whitt's summation bound is that the difficult solution of the fixed point equations can be avoided, and if the network is only moderately loaded the accuracy can still be sufficient. However, the higher the load is, the larger are the inaccuracies.

This method can easily be generalised for multirate networks:

$$L_r^{exact} \leq \sum_{m,l} \tilde{B}_{ml} A_{mlr} \quad (10)$$

where  $\tilde{B}_{ml}$  stands for the blocking probability of traffic of class  $m$  over link  $l$  evaluated without any load reduction.

## 5 Multicast Tree Blockings

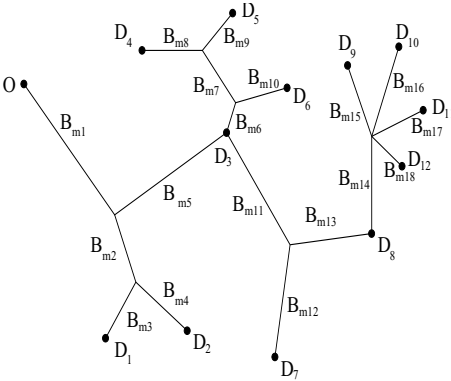
Multicasting plays increasingly important role in ATM and IP networks [20].

### 5.1 Reduced Load and Link Independence

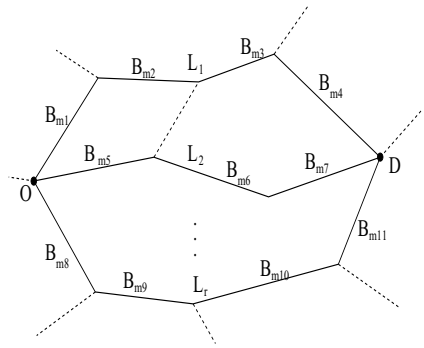
The classical *reduced load approximation under link independence assumption* can be generalised for approximating the connection-blocking probabilities of point-to-multipoint connections (multicast trees) in both single- and multi-rate loss networks.

In modern networks point-to-multipoint connections are increasingly often required. Denote that all multipoint-to-multipoint connections can be built up from point-to-multipoint connections. For example for  $K$  nodes a multipoint-to-multipoint connection can be built up as  $K$  one-point to  $(K - 1)$ -point connections. It is much more efficient to use point-to-multipoint trees (Steiner-tree problem) where the information is duplicated (or multiplicated) only in the tree-branching points instead of using direct point-to-point connections. In this case the total resource requirement can be significantly reduced.

The blocking probability for establishing a point-to-multipoint connection can be approximated in analogous way to that one presented in Section 4. The point-to-multipoint connection is considered blocked if more then  $k$  among  $K$  end-nodes can not be connected where  $k < K$ . This case is called  $k$ -reliable multicast. When  $k = K$  the multicast is called fully reliable. Approximations for connection blocking probabilities will be given, first for fully reliable multicast, then for the  $k$ -reliable case.



**Fig. 1.** Dependence of Link and Tree Blocking Probabilities



**Fig. 2.** Dependence of Link, Route and OD-pair Blocking Probabilities

The formulation is as follows:

The events of call-blocking on individual links used by the multicast tree are assumed to take place independently and the load offered to link  $j$  is assumed to

consists not of the total load offered but only the proportion of traffic not blocked by other links. To be more precise let  $B_{m,j}$  denote the blocking probability of traffic class  $m$  on link  $j$ , and let  $E_{m,j}(\rho_1, \dots, \rho_M; C_j)$  be the blocking function which gives the blocking probability of traffic class  $m$  on link  $j$  when the offered class  $m$  bandwidth demand on link  $j$  is  $\rho_m$  and link capacity is  $C_j$ . Then

$$B_{mj} = E_{mj}(\rho_{1j}, \dots, \rho_{Mj}; C_j) \quad \text{for all } m \text{ and } j \quad (11)$$

$$\lambda_t = \nu_t(1 - L_t) \quad \text{where} \quad (1 - L_t) = \prod_{m,l} (1 - B_{ml})^{A_{mlt}} \quad (12)$$

$$\rho_{mj} = d_{mj} \sum_t (1 - B_{mj})^{-1} \lambda_t A_{mjt} = d_{mj} \nu_{m,j} \quad \text{for all } m \text{ and } j \quad (13)$$

where  $\nu_{m,j}$  is the offered class  $m$  traffic to link  $j$  under the reduced load approximation. Denote that instead of routes  $r$  we calculate all values for multicast trees  $t$ .

To obtain the *point-to-multipoint* connection blocking probabilities we should use the above formula but for all links the multicast tree  $t$  consists of.

$$L_t^{exact} \approx 1 - \prod_{m,l} (1 - B_{ml})^{A_{mlt}} \quad (14)$$

The link blocking probabilities  $B_{ml}$  are given as a solution to (11), (12) and (13) and can be obtained by, e.g., repeated substitution.

This approach can be further generalised for multipoint-to-multipoint connections in analogous way. Furthermore, formula for  $k-1$  reliable multicast can be derived as follows:

$$L_t^{exact} \approx 1 - \prod_{m,l} (1 - B_{ml})^{A_{mlt}} - \sum_{\forall j \supset D} B_{mj} \prod_{\forall l \in t \setminus j} (1 - B_{ml})^{A_{mlt}} \quad (15)$$

where  $\forall j \supset D$  stands for all those links which connect the leave nodes to the multicast tree.

By counting the number of nodes accessible via link  $j$  in the multicast tree, weight  $w_j$  can be assigned to links  $j$ . Then, blocking dependence for a  $k$ -reliable multicast can also be determined, where  $k = (K-2), (K-3), \dots, 1$  and  $K$  is the number of end-points of the multicast tree.

$$L_t^{exact}(k) \approx 1 - \sum_{\forall \mathbf{p} \in \{0,1\}^N \text{ where } \biguplus p_q w_j < K-k} \prod_{q=1}^N B_{mq}^{p_q} (1 - B_{mq})^{\bar{p}_q} \quad (16)$$

where  $\biguplus p_q w_j$  stands for the number of blocked endpoints of the multicast tree. That is, if the connection from the originator  $O$  to destination node  $D_i$  is blocked by two or more simultaneous link blockings along that path it is counted once only.  $N$  is the total number of links the tree consists of.  $\mathbf{p} = (p_1, p_2, \dots, p_q, \dots, p_N)$  and  $\bar{p}_q$  stands for the negation of  $p_q$  (i.e., if  $p_q = 1 \Rightarrow \bar{p}_q = 0$ , and if  $p_q = 0 \Rightarrow \bar{p}_q = 1$ ).

## 5.2 Whitt's Summation Bound

Formula (9) can be used to bound both, route and tree blocking probabilities for a multicast tree in way as described above for the case of reduced load approximation. The blocking probability for a node-pair connected along route  $r$  can be upper bounded as follows:

$$L_r^{exact} \leq \sum_{m,l} \tilde{B}_{ml} A_{mlr} \quad (17)$$

where  $r \subset t$ , or for the case of the whole point-to-multipoint connection (i.e., multicast tree  $t$ ) can be upper bounded by:

$$L_t^{exact} \leq \sum_{m,l} \tilde{B}_{ml} A_{mlt} \quad (18)$$

## 6 OD-Pair Blocking

### 6.1 The Induced Load (IL) Approximations under the Route Independence Assumption

When Alternative Routing (AR) is applied in a network a connection setup attempt is considered blocked not after the first unsuccessful trial, but after  $R$  retrials. The proposed OD-pair blocking approximations are valid for the case when the number of alternative routes  $k$  of one OD-pair is equal to the number of alternative routes  $|R|$  for the SAR alternative routing strategy and for any  $k \leq |R|$  in case of DAR.

This approximation violates the assumption on Poissonian character of the offered traffic, because the overflow traffic becomes more peaked (bursty). For this reason the accuracy of the proposed approximation will deteriorate as the amount of the overflow traffic increases. This is valid for Alternative Routing techniques in general. However, for Dynamic Alternative Routing (DAR) the overflow is dynamically minimised, since after each unsuccessful connection establishment attempt the sequence of trials is updated, i.e., the route with high blocking probability will receive less traffic. For this reason the accuracy of the proposed approximation for DAR is excellent at moderate and low loads.

First, the approximation for Sequential AR (SAR) will be described, then that one for DAR. For the all OD-pair blocking probability approximations *Route Independence* is assumed, i.e., that the event of blocking a connection setup over one route does not influence the event of blocking its retrial over another route. The term *Induced Load* means, that the blocking of one route induces more load, i.e. the traffic will be re-offered to another route.

### 6.2 OD-Pair Blocking Approximation for SAR

The sequence of alternative routes is static. The routes are tried one-by-one in the given order until the call is established or all alternative routes exhausted.

If all routes are unable to accommodate the connection it is considered blocked. The procedure of approximating OD-pair blocking probabilities is as follows:

1. Set the iteration counter  $i = 0$  and the initial route-blocking probabilities  $L_r$  of routes  $r$  to  $L_r = 0.1$ ,  $\forall r \in R_o$ ,  $\forall o \in O$  ( $R_o$  is the ordered set of all routes of the OD-pair  $o$ ;  $O$  is the set of all OD-pairs within the network). Let  $\nu_o$  be the traffic offered to OD-pair  $o$  for all OD-pairs  $o \in O$ .  $L_0^{(i)} = 1 \forall o \in O$  in all iterations  $i$ .
2.  $i++$ ;
3. Offer traffic  $\nu_o^{(i)}(r)$  to route  $r$  of the OD-pair  $o$ , where routes are sorted from 1 to  $|R_o|$  according to the order of trials, where  $|R_o|$  stands for the cardinality (number of elements) of  $R_o$ .

$$\nu_o^{(i)}(r) = \nu_o \prod_{q=0}^{r-1} L_q^{(i-1)} \quad \forall r \in R_o, \forall o \in O \quad (19)$$

4. Determine the blocking probabilities  $B_{ml}$  of all links  $l$  of the network for all traffic classes  $m$  and that of all routes  $r$  ( $L_r$ ) for all OD-pairs  $o$  iteratively using, e.g., the RL approximation as described in Sections 3 and 4.
5. Express the OD-pair blocking probability  $B_o$  for all OD-pairs  $o$  as

$$B_o^{(i)} = \prod_{\forall r \in R_o} L_r^{(i)} \quad (20)$$

where  $R_o$  is the number of both, routes and retrials for OD-pair  $o$ .

6. Goto Step 2 until the OD-pair blocking probabilities reach their stationary value, which is known to exist by Brouwer's fixed point theorem.

Iterative repetition of steps 2 and 3 models spreading the traffic between different routes, i.e., it ensures both, the overflow of the traffic from the first route tried to others within one OD-pair as well as the interactions between OD-pairs, since the routes of different OD-pairs share common links. The speed of convergence depends on both the average and maximal number of routes  $|R_o|$  per OD-pair.

### 6.3 OD-Pair Blocking Approximation for DAR

The first path to be tried will always be the last path which has successfully accommodated the call. There are two alternatives for determining the sequence of alternative routes. First, the sequence is dynamically changing according the route-blocking probabilities experienced (DAR1). Second, after an unsuccessful trial a new route is randomly chosen (DAR2). The DAR1 strategy will be first investigated. Its dynamic behaviour can be modelled by the following iterative method proposed.

1. Set the iteration counter  $i = 0$  and the initial OD-pair blocking probability  $B_o^{(i)} = 1$ .

2. For  $\forall r \in R_o$ ,  $\forall o \in O$  set  $\nu_o^{(0)}(r) = \frac{\nu_o}{|R_o|}$  and determine  $L_r^{(0)}$  by, e.g., RL approximation.
3.  $i \leftarrow i + 1$ ;
4. Offer traffic  $\nu_o^{(i)}(r)$  to the  $r^{th}$  route of the OD-pair  $o$  in the  $i^{th}$  iteration

$$\begin{aligned} \nu_o^{(i)}(r) = & \nu_o \left( \frac{1 - L_r^{(i-1)}}{\sum_{\forall z \in R_o, p_z=0} (1 - L_z^{(i-1)})} + \right. \\ & \left. + \sum_{\forall \mathbf{p} \in \{0,1\}^{|R_o \setminus r|}} \frac{1 - L_r^{(i-1)}}{\sum_{\forall z \in R_o} p_z (1 - L_z^{(i-1)})} \prod_{\forall q \in R_o \setminus r} \frac{(1 - L_q^{(i-1)}) (L_q^{(i-1)})^{p_q}}{\sum_{\forall z \in R_o \setminus r} (1 - L_z^{(i-1)})} \right) \end{aligned} \quad (21)$$

where  $\mathbf{p} = (p_1, p_2, \dots, p_q, \dots, p_{R_o})$ , and  $\nu_o$  is the traffic originally offered to OD-pair  $o$ . The first term of (21) expresses that the load will be offered to route  $r$  proportionally to its availability (normalised for all available paths). The second term expresses that all traffics blocked at other routes will be offered with certain probability (first factor of the second term) to route  $r$ . Assuming that route-blocking probabilities are small ( $L_q \approx 0$ ), the relation can be simplified:

$$\nu_o^{(i)}(r) = \nu_o \frac{1 - L_r^{(i-1)}}{\sum_{\forall z \in R_o} (1 - L_z^{(i-1)})} \left( 1 + \sum_{\forall q \in R_o \setminus r} \frac{(1 - L_q^{(i-1)}) L_q^{(i-1)}}{\sum_{\forall z \in R_o \setminus r} (1 - L_z^{(i-1)})} \right) \quad (22)$$

When determining the offered traffic to every single route  $r$  (according to (22)) we implicitly assume that each unsuccessful trial is retried once only. This slightly underestimates the amount of traffic offered to routes  $r$  and therefore the whole OD-pair blocking probability.

5. Determine the blocking probabilities of all links and then  $L_r^{(i)}$  of all routes  $r$  of all OD-pairs  $o$  using, e.g., the RL approximation (Section 4).
6. Express the OD-pair blocking probability  $B_o^{(i)}$  for all OD-pairs  $o$  as

$$B_o^{(i)} = \prod_{\forall r \in R_o} L_r^{(i)} \quad (23)$$

where  $R_o$  is the number of both, alternative routes and retrials for OD-pair  $o$ .

7. Go to Step 3 until blocking probabilities of OD-pairs reach their stationary value, i.e., while  $\exists o \in O$ ,  $|B_o^{(i-1)} - B_o^{(i)}| > \epsilon$

When this routing strategy is applied in a network the convergence of the approximation is slower than that for SAR because of iterations needed, but the accuracy is better, since the overflow traffic is minimised.

It should be mentioned, that approximation (22) is equivalent to assuming, that after an unsuccessful first attempt one retrial is allowed only, which always leads to success. According to DAR1 technique frequency of choosing a route  $r$

among  $R_o$  alternatives is proportional to the probability that route  $r$  will not block the call, i.e., to  $1 - L_r$ . Assuming that after an unsuccessful first attempt  $k - 1$  retrials are allowed ( $k$  trials in total), where  $k < |R_o|$ , approximation (23) can be written as:

$$B_o^{(i)}(k) = \sum_{\forall j_1 \in R_o} \frac{1 - L_{j_1}}{\sum_{\forall l_1 \in R_o} (1 - L_{l_1})} L_{j_1} \left( \sum_{\forall j_2 \in R_o \setminus j_1} \frac{1 - L_{j_2}}{\sum_{\forall l_2 \in R_o \setminus j_1} (1 - L_{l_2})} L_{j_2} \dots \right. \\ \left. \dots \left( \sum_{\forall j_k \in R_o \setminus \{j_1, j_2, \dots, j_{(k-1)}\}} \frac{1 - L_{j_k}}{\sum_{\forall l_k \in R_o \setminus \{j_1, j_2, \dots, j_{(k-1)}\}} (1 - L_{l_k})} L_{j_k} \right) \dots \right) \quad (24)$$

It can be easily shown, that  $B_o^{(i)}(k)$  is equal to (23) for  $k = |R_o|$ .

For the case of DAR2 the first trial will have the same probability as described for DAR1, but the second trial will be distributed between available (not yet tried) alternative paths with equal probabilities. For DAR2 when  $k < |R_o|$ , approximation (24) can be written as:

$$B_o^{(i)}(k) = \sum_{\forall j_1 \in R_o} \frac{1 - L_{j_1}}{\sum_{\forall l_1 \in R_o} (1 - L_{l_1})} L_{j_1} \left( \sum_{\forall j_2 \in R_o \setminus j_1} \frac{1}{|R_o| - 1} L_{j_2} \dots \right. \\ \left. \dots \left( \sum_{\forall j_k \in R_o \setminus \{j_1, j_2, \dots, j_{(k-1)}\}} \frac{1}{|R_o| - k + 1} L_{j_k} \right) \dots \right) \quad (25)$$

For DAR2 it can be also easily shown, that  $B_o^{(i)}(k)$  is equal to (23) for  $k = |R_o|$ .

## 7 Conclusion and Future Work

In this paper we have proposed new approximations for analytical evaluation of call blocking probabilities for some alternative routing techniques with different number of alternatives and retrials, as well as for full- to k-reliable multicast connections. The latter is generalised for multipoint-to-multipoint connections. The obtained results are supported by simulation results. We have shown, that the fixed routing with point-to-point connections and with no retrial is a special case of our problem, and our formula is than reduced to the reduced load approximation with link independence assumption.

The importance of these results lies in the fact that in modern telecommunication networks the number of links is much smaller than the number of routes. For network operators, link blocking values are often available and in case of reconfiguration the fast and simple route and OD pair blocking calculations can be performed.



## References

1. A. Girard, *"Routing and Dimensioning of Circuit Switched Networks"*, Addison-Wesley, 1990
2. F.P. Kelly, *"Routing in Circuit-Switched Networks: Optimization, Shadow Prices and Decentralization"*, Advanced Applied Probability, Vol. 20, pp. 112-144, 1988
3. L. Ast, S. Blaabjerg, T. Cinkler, *"Approximations for Revenue Optimization in Multi-Rate Loss Networks"*, ITC Seminar: New Telecommunication Services for Developing Networks, St.-Petersburg, June 1995
4. T. Cinkler, G. Fodor, L. Ast, S. Racz, *"End-To-End Blocking Probability Approximations for Resource Management in Multirate Loss Networks"*, International Teletraffic Congress Seminar, Bangkok, 1995
5. A. Girard, *"Routing and Dimensioning in Circuit Switched Networks"*, Addison Wesley Publishing Company, ISBN 0-201-12792-X, 1990
6. J.W. Roberts, Z. Dziong *"Congestion Probabilities in a Circuit Switched Integrated Services Network"*, North-Holland, Performance Evaluation 7 267-284, 1987
7. K.W. Ross, *"Multiservice Loss Models for Broadband Telecommunication Networks"*, Springer Verlag, ISBN 3-540-19918-7, 1995
8. M. Ritter, P. Tran-Gia, editors, *"Multi-Rate Models for Dimensioning and Performance Evaluation of ATM Networks"*, Cost 242 Interim Report, June 1994
9. T. Cinkler, G. Fodor, L. Ast, S. Racz, *"End-to-End Blocking Probability Approximations for Resource Management in Multi-Rate Loss Networks"*, ITC Seminar, Bangkok, November 1995
10. T. Cinkler, G. Fodor, S. Racz, T. Henk, *"Simulative Analysis of End-to-End Call Blocking Probability Evaluation Techniques in Multi-Rate Loss Networks"*, 10th European Simulation Multiconference, Budapest, June 1996
11. L. Ast, S. Blaabjerg, T. Cinkler, G. Fodor, S. Racz, *"Blocking Probability Approximations and Revenue Optimization in Multi-Rate Loss Networks"*, Special Issue of "Simulation", Modelling and Simulation of Computer Systems and Networks: Part One, Vol.68, No.1, Jan 1997
12. R. Guérin, H. Ahmadi, M. Naghshineh, *"Equivalent Capacity and Its Application to Bandwidth Allocation in High Speed Networks"* IEEE JSAC, September 1991
13. Z. Dziong, J.W. Roberts, *"Congestion Probabilities in a Circuit Switched Integrated Service Network"*, Performance Evaluation 7, 1987
14. Cost 242 Interim Report, M. Ritter, P. Tran-Gia (Ed.), *"Multi-Rate Models for Dimensioning and Performance Evaluation of ATM Networks"*, Commission of the European Communities, June 1994
15. L. Delbrouck, *"On the Steady-State Distribution in a Service Facility Carrying Mixtures of Traffic with Different Peakedness and Capacity Requirements"*, IEEE Trans. on Comm., No. 11, 1983
16. S-P. Chung, K.W. Ross, *"Reduced Load Approximations for Multirate Loss Networks"*, ACM/IEEE Transactions on Networking, 1993, pp. 1222-1231
17. A. Faragó, S. Blaabjerg, L. Ast, T. Henk, *"A New Degree of Freedom in ATM Network Dimensioning: Optimizing the Logical Configuration"*, IEEE JSAC, Journal on Selected Areas in Communications, 1996
18. F.P. Kelly, *"Blocking Probabilities in Large Circuit-Switched Networks"*, Adv. Appl. Prob. Vol. 18, 1986, pp. 473-505
19. W. Whitt, *"Blocking When Service Is Required From Several Facilities Simultaneously"*, AT&T Technical Journal Vol. 64, No. 8, pp. 1807-1856, October 1985
20. S. Paul, *Multicasting on the Internet and its Applications*, 1998, Kluwer, ISBN 0-7923-8200-5

# Consolidation Algorithm Interoperability in Point-to-Multipoint ABR Services in ATM Networks

Naris Rangsinoppamas<sup>1</sup>, Tanun Jaruvitayakovit<sup>1</sup>, Wisitsak Sa-niamsak<sup>2</sup>,  
Praphan Pavarangkoon<sup>2</sup>, and Prasit Prapinmongkolkarn<sup>1</sup>

<sup>1</sup> Telecommunication System Research Laboratory  
Department of Electrical Engineering, Faculty of Engineering  
Chulalongkorn University, Bangkok, Thailand  
`nrr@kmitnb.ac.th`

<sup>2</sup> Telecommunication System Research Laboratory  
Department of Electrical Engineering, Faculty of Engineering  
King Mongkut Institute of Technology North Bangkok, Bangkok, Thailand

**Abstract.** The point-to-multipoint ABR service is a very important service in ATM networks. Recently, several researchers have studied the issues on multicast ABR services and have proposed a family of consolidation algorithms. All of the previous proposed algorithms have some drawbacks such as slow response, high consolidation noise and high implementation complexity etc. Hence, we propose a new efficient consolidation algorithm which overcomes all such drawbacks. We also recognize that in point-to-multipoint ATM networks, there is a possibility that the branch points will adopt different consolidation algorithms. We, therefore, investigate the interoperability issue and the impact of using different types of consolidation algorithm at the upper and lower stream branch points and show that the branch points with different consolidation algorithms can work interoperably. It is concluded from the simulation results that in order to avoid the consolidation noise and get a faster response in the network the most upper stream branch point (the nearest one to the source) has to be implemented with a high accuracy consolidation algorithm and the lower stream branch point(s) should be implemented with a fast response consolidation algorithm. Moreover, using our proposed consolidation algorithm, fast response, low consolidation noise and low implementation complexity are achieved.

## 1 Introduction

The control of ABR service in ATM network is inherently closed loop [1]. The sources send the data at the rate specified by the feedback received from the networks or destinations. There have been many rate-based congestion control schemes proposed ranging from a single bit congestion indicator [2] to explicit rate feedback to the source [3]. All of these schemes were proposed for point-to-point ABR service. In an explicit rate-based control scheme the source generates

the Resource Management cells (RM) to the destination. We called these cells as forward RM cells (FRM). The destination used this RM cells to feedback to the source an Explicit Rate (ER) indicating an available bandwidth on the bottlenecked path connecting to it. These RM cells are called backward RM cells (BRM). The source is controlled to a minimum rate between the rate specified in BRM cell and the preset value at the source. In ATM point-to-multipoint connections, the major concern at a branch point is how to consolidate the BRM cells from destinations to the source. The branch point that wait for the BRM cells from all destinations in the multicast tree will introduce a slow response or a consolidation delay. On the other hand, if the branch point consolidates some or even a single BRM cell from the destinations and then returns to the source. This make it has a fast response but will lead to a consolidation noise because the returned BRM cell contains incomplete congestion information for some downstream branches. Hence, the challenge of designing rate-based control algorithm in ABR point-to-multipoint communications is how to meet the different requirements requested from multiple destinations i.e. a fast response, a low consolidation noise etc. with the least expenses of an accuracy and complexity. In part of this paper, we proposed a new efficient consolidation algorithm. The algorithm offers a faster response, less consolidation noise, better link utilization and less complexity. The details of the algorithm will be explained in Section 3. There were several papers proposing the consolidation algorithm. The consolidation algorithm can be roughly categorized into two types the slow response with low noise and the fast response with high noise. To our best knowledge, there is no paper about interoperability issue in point-to-multipoint ABR services has been presented. We are aware that there is a possibility that the branch points will adopt different consolidation algorithm. Consequently, in this paper we investigate the inter-operability issue and the impact of using different types of consolidation algorithm at the upper and lower stream branch points.

The organization of this paper is as follows. In Section 2, we review previously proposed related works. Our proposed consolidation algorithm is described in Section 3. Simulation results and discussions of the proposed algorithm and the interoperability issue are presented in Section 4. The paper's conclusions are drawn in the Section 5.

## 2 Related Works

There are several consolidation algorithms proposed in the previous works [4]-[11]. A Minimum Explicit Rate (MER) register and MCI and MNI flags are widely used in the previously proposed algorithms. MER is used to temporarily store the minimum of the ER's values among those indicated by the BRM cells received from the branches and the ER calculated locally at the branch point. As mentioned earlier, we can categorize the consolidation algorithm into two types.

1. **The wait-for-all consolidation algorithm.** The algorithm has been proposed in [6]. In addition to MER, MCI and MNI, the algorithm requires two more counters: Number\_of\_BRMreceived and Number\_of\_Branches. Num-

ber\_of\_BRMreceived counts the number of branches from which BRM cells have been received at a branch point(after the last BRM was sent by the branch point)and Number\_of\_Branches stores the number of branches of a point-to-multipoint at this branch point. A branch point waits for BRM cell from all branch. When the value of this two counters get equal then the branch point returns a BRM cell with the completed congestion information to its upstream node (or source) whenever it receives a first FRM cell after completely consolidating the BRM cells. This algorithm has a slow response but does not introduce a consolidation noise.

**2. The not wait-for-all consolidation algorithm.** The algorithms have been proposed in [5]. The basic idea of these algorithms is that a branch point calculates ER, CI and NI value according to its local congestion level, updates MER, MCI and MNI and returns a BRM cell to its upstream node whenever it receives a FRM cell. This makes the branch point feedbacks the congestion information faster than the first type, however, because of lack of the complete congestion information the consolidation noise is quite high.

There are some proposed algorithms to achieve a fast response with low consolidation noise [9]. The main idea is the branch point has to wait for all BRM cells from all branches like the algorithm proposed in [6]. If there is an overloaded condition say, the rate in current BRM cell is much less than last BRM cell, exists at the branch point. The algorithm can detect this phenomena and a branch point immediately feedbacks the BRM cell with the low ER value to the source to avoid data overflow. The branch point will send the BRM back again when BRM cell from all branches have been arrived. With this approach, we notice that the branch point seems in-active during waiting for all BRM cells. Hence, this will slow down the branch point's response. Moreover, if there is more available bandwidth, for example, from the off period of VBR traffic during this interval. This bandwidth can not be utilized by the network resulting in lower network utilization.

### 3 Selective BRM Feedback Consolidation Algorithm

In this section, an efficient consolidation algorithm is presented. We have learnt from the previously proposed papers that if the branch point does not wait for all BRM cells it will introduce the noise. On the other hand, the response is very slow if we wait for all. Based on the fact that the branch point has to send the least ER value among all BRM cells from its branches to the source. We, hence, design a consolidation algorithm called Selective BRM Feedback (SBF) that feedbacks the BRM cell to the source selectively. The branch point does not wait for all BRM cells but selects the BRM cell which contains the least ER value to send to the source. The SBF is designed to achieve a fast response, low consolidation noise and low complexity. It uses 3 registers at the branch point. A MER register is used for storing the ER value, a Branch\_number register is used for storing the number of branch from which the branch point receives the BRM cell and a Balance\_RM is used for controlling the BRM to FRM ratio to

be 1. The algorithm can take care of overload condition in downstream branches and can utilize, during the transient state, the available bandwidth, especially left from the VBR source in the network, without consolidation noise. It is also insensitive to the number of branches and branch point level in the network. The algorithm works as follows.

Upon the receipt of the BRM cell, the branch point checks the branch number and the ER value. If the branch number is different from the current branch number stored in Branch\_number register and the received ER value is less than the current ER value stored in MER register, the MER and Branch\_number are updated to the new values. Otherwise, the received BRM cell is discarded. In case BRM cell comes from the same branch number stored in the Branch\_number register, the branch point always updates the ER value in MER, no matter it is more or less with respect to the previous one. Thus, these registers always preserve the latest congestion information in the downstream branches. From the above explanation, alternatively, we can show in four cases.

Case 1: if (ER from BRM < MER and Branch\_Number = j) → Update MER and Branch\_Number

Case 2: if (ER from BRM > MER and Branch\_Number = j) → Update MER and Branch\_Number

Case 3: if (ER from BRM < MER and Branch\_Number ≠ j) → Update MER and Branch\_Number

Case 4: if (ER from BRM > MER and Branch\_Number ≠ j) → Not update MER and Branch\_Number

We can see that in case 1 and case 3 (ER from BRM < MER) the MER is always updated. This is designed to feedback the lower value of ER from BRM quickly to the source. This technique statistically reduce the waiting time of the branch point i.e. if the branch that has the lowest ER is located nearest to the branch point, the branch point has not to wait for consolidating BRM from all branches. In case 2, where the value of ER from BRM is larger than MER and the Branch\_Number is equal to j. The branch point is also send the updated ER to source. This is our intention to let the branch point to has feature that can utilize the bandwidth left in this lowest branch. For example, if this branch share the bandwidth with the VBR traffic and in this epoch the VBR traffic is active. Hence, the traffic available for ABR service in this branch is the traffic that left from VBR. For the next epoch (normally equal to the Average Interval (AI) time in ERICA), if the VBR traffic is off. The available bandwidth for ABR is increased and branch point can recognize this change and feedback to the source to increase its rate. For case 4, the branch point discards the BRM cell because branch number j being the branch that can support the lowest rate (has the least bandwidth). The branch point will be suffered from the buffer overflow if we update the MER with the ER from BRM from the branches that has a higher rate. With this approach, the branch point is updated by the BRM cell from the most congested branch and sends it back to the source promptly without waiting for all BRM cells from all branches (that some of them might be non-responsive). The salient features of SBF are a fast response of branch point while a low consolidation noise condition is preserved. In addition, it has a low

---

```

Upon the receipt of forward RM(ER, CI, NI) cell:
  1. Multicast this RM cell to all participating branches
  2. Let Balance_RM = Balance_RM + 1

Upon the receipt of backward RM(ER, CI, NI) cell from branch j:
  IF (ER from BRM_MER or Branch_number == j) THEN
    IF (ER from BRM == MER && Balance_RM <= 0) THEN
      Discard RM cell;
    ELSE
      MER = ER from BRM;
      Branch_number = j;
      ER = min(ER from BRM, minimum ER calculated by rate allocation
               algorithm for all branches);
      CI = CI from BRM, and NI = NI from BRM;
      Send RM(ER, CI, NI) cell back to source;
      Let Balance_RM = Balance_RM - 1;
    ELSE
      Discard RM cell;

```

---

**Fig. 1.** Pseudo code of the proposed algorithm

implementation complexity (used only 3 registers while the previously proposed algorithms used more registers and some additional flags) The pseudo code and flow chart of the new consolidation algorithm are shown in Fig. 1 and Fig. 2.

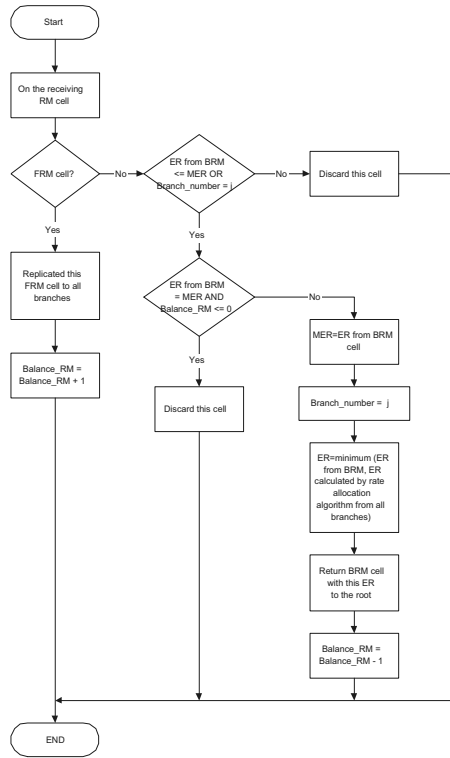
## 4 Simulation Results

For convenience, all algorithms proposed in [5], [6], [9] will be referred simply to the authors' name. To refer to our proposed algorithm, we will use the term 'SBF'.

### Parameters Setting

We use these values in both Network Model I and Network Model II

1. Except where indicated, all link capacity are 150 Mbps.
2. All switch-to-end system links are 50 km except where specified.
3. The switch target utilization is 90%.
4. The switch averaging interval is set to 1 ms.
5. The parameter Transient Buffer Exposure (TBE) is set to large values for preventing the rate decrease.
6. The source parameter Rate Increase Factor (RIF) is set to 1.
7. Simulation time is 200 ms, and 300 ms for Network Model I and Network Model II, respectively.
8. The VBR source bursts for 100 ms and silence for 150 ms.
9. The Peak Cell Rate (PCR) and the Initial Cell Rate (ICR) are 150 Mbps
10. Use ERICA switch algorithm [12] as rate allocation algorithm.



**Fig. 2.** Flow chart of the proposed algorithm

#### 4.1 Proposed Consolidation Algorithm

In this section, we will show the simulation results of the SBF. In [9], it was shown that Fahmy is better than Roberts and Ren. Hence, we compare SBF to Fahmy only. (We actually compare SBF to Roberts and Ren. The results show that SBF has much better performance than the others in terms of response time and consolidation noise but did not show the results here because we intend to focus on the interoperability issue). The network configuration is shown in Fig. 3. Source S1 is a point-to-multipoint connection and its destinations are dS1, dS2 and dS3. S4 and VBR source is a point-to-point connection and has a dS4 and dVBR destination, respectively. For convenience, we would separate the response of the network into two intervals. First interval lies between 0-80 ms and the second is from 80 ms onwards. During the first interval, the BRM cells from all destinations except for dS3 have already arrived at S1. This is because the BRM cell from dS3 has a Round Trip Time around 81 ms. (we use 5 microseconds per kilometer delay so the RTT for dS3 is about 81 ms). In Fig. 4(a), the BRM cell from dS2 is arrived at S1 at 6.5 ms. We can see that the Allowed Cell Rate (ACR) of S1 for both SBF and Fahmy is reduced from 150 Mbps to a fair share value of available bandwidth left in Link2. After this point, SBF can tell S1 to send a data according to an available bandwidth left from

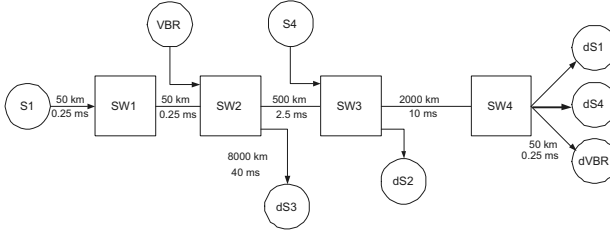


Fig. 3. Network Model I

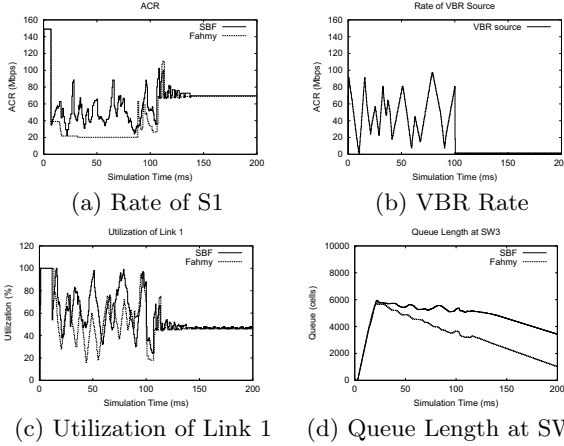


Fig. 4. Simulation results of Network Model I

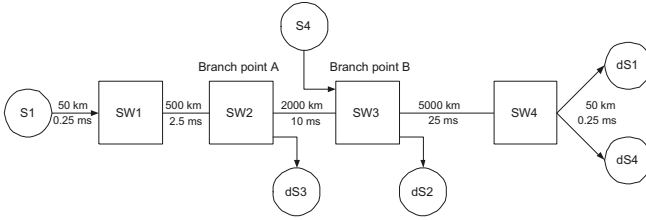
being used by the VBR source (Fig. 4(b)) while S1 of Fahmy can not. Instead, Fahmy suppresses the ACR of S1 to nearly constant low value. This leads to the lower utilization of Link1 for Fahmy comparing to SBF during the transient state (Fig. 4(c)). However, the average queue length of the SBF is a bit higher than Fahmy (Fig. 4(d)).

During the second interval, it seems no big difference between the two algorithms. The reason behind this is all BRM cells from every destinations have already reached at SW2 which mean the BRM cells from all branches are fully distributed in their path. Hence, if there is another VBR burst (after 100 ms), both algorithms can detect and update the ER field in the current RM cell and send it back to S1 immediately.

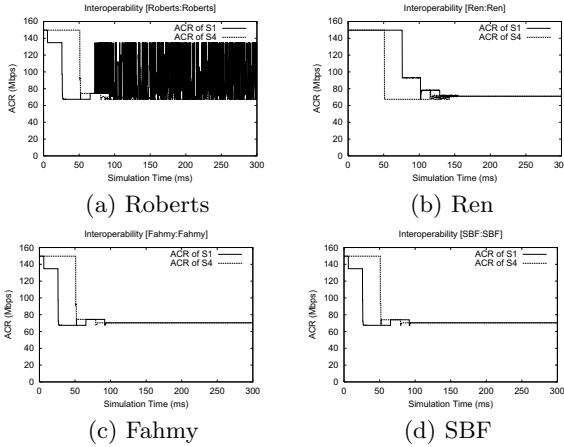
## 4.2 Interoperation

In this section, We investigate the interoperation by using both the same consolidation algorithm and different consolidation algorithms at the branch points. The Network Model II is shown in Fig. 5. S1 is a source for a point-to-multipoint connection and its destinations are dS1, dS2 and dS3. The Round Trip Time (RTT) from S1 to dS1, dS2 and dS3 are 76 ms, 26 ms and 6 ms, respectively. S4 is a source for point-to-point connection. The RTT from S4 to its destination,





**Fig. 5.** Network Model II



**Fig. 6.** ACR of the source S1 and S4 using the same consolidation algorithm

dS4, is 51 ms. SW1 and SW 4 are the switching nodes and SW2 and SW3 are the branch points. The algorithms which have been described in Section 2 and Section 3 are implemented at Branch point A (SW2) and Branch point B (SW3). Branch point A is considered a upper stream branch point while Branch point B is a lower stream branch point. S1 and S4 are set to initially start sending with ICR value. This setting, on one hand, causes a high network utilization, on the other hand, exhibits a congestion and large queue length at Branch point B. To avoid the congestion, S1 and S4 should reduce their Allowed Cell Rate (ACR) to a fair share rate as fast as they can. However, the reduction of the ACR could not be occurred until receiving a turn-around RM cell from its destinations.

Fig. 6 shows the results that Branch point A and Branch point B are implemented with the same consolidation algorithm. The results of branch points that are both implemented with Roberts is shown in Fig. 6(a). Roberts is the fast response but high consolidation noise algorithm. The ACR of S1 is reduced promptly from 150 Mbps (ICR) to 135 Mbps (90% of PCR as preset at the switch) when the first BRM cell from dS3 arrived at S1 (at 6 ms time). This ACR value lasts until the first BRM cell from dS2 arrived at S1 (at 26 ms time). The ACR is reduced again to 67.5 Mbps (the fair share bandwidth between S1 and S4 at Branch point B). After the arrival of the farthest BRM (from dS3) at 76 ms, S1 start to oscillate and the consolidation noise is occurred. This is

**Table 1.** Simulation results for a network using the same consolidation algorithm at Branch point A and Branch point B

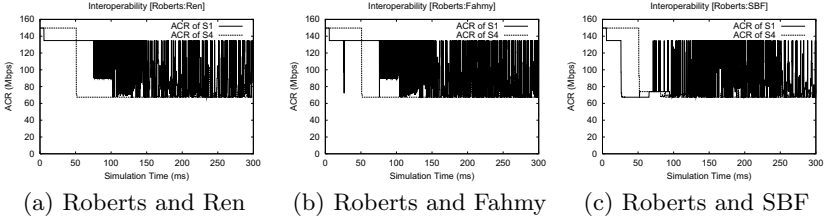
Branch point A	Branch point B	Consolidation algorithm complexity	Consolidation noise	Time to get to fair share rate (ms)
Roberts	Roberts	Low	Yes	-
Ren	Ren	Low	No	$\geq 76$
Fahmy	Fahmy	Medium	No	26
SBF	SBF	<Medium	No	26

because the ACR of S1 depends on the BRM cell being sent to it by the Branch point A. If, at that moment, the BRM cell comes from dS3 then S1 is set ACR to 135 Mbps. On the other hand, if BRM cell comes from dS1 or dS2 then ACR of S1 will be set to 67.5 Mbps. Hence, the ACR will oscillate between 67.5 Mbps and 135 Mbps. For S4, the ACR is dropped to a fair share value after the RTT is reach (51 ms). Because S4 is a unicast connection, it results the same graph for all figures in this section. The result of implementing Ren, which is a 'wait-for-all' BRM algorithm, at the branch points is shown in Fig. 6(b). We see that there is no consolidation noise. However, the ACR of S1 is dropped after BRM cell from dS1 (the farthest) has arrived. It takes 76 ms before starting to get approach a fair share rate. Comparing this slow response to Roberts' fast response time (6ms and 26 ms), it is about 50 ms different in response time and the difference in ACR of S1 during this 50 ms period is 82.5 Mbps (PCR minus fair share rate). This will cause an approximately 4.125e6 bit or 9730 excessive cells built up at Branch point B and cells may be lost if the buffer is insufficiently provided. For Fahmy and SBF, the results are shown in Fig. 6(c) and 6(d), respectively. We see that both algorithms perform identically in this network model. They exhibit a fast response and no consolidation noise, however, SBF has less in implementation complexity. Table 1 summarizes the simulation results for the network using the same consolidation algorithm at the branch points.

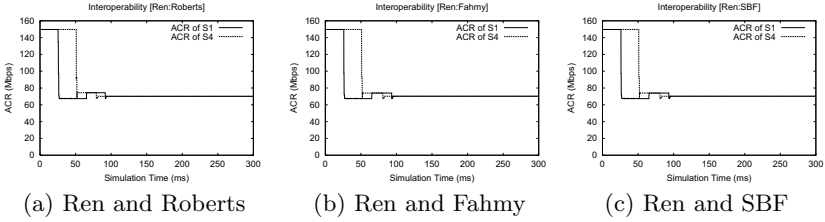
Fig. 7 to Fig. 10 are the simulation results of the interoperation of the branch points implementing with different consolidation algorithms. We investigate all possible combinations of the algorithms. The notation e.g. Roberts and Ren we used below each graph in all figures means Branch point A and Branch point B are implemented with Roberts and Ren, respectively. We can comprehensively analyze the results as follows;

1. Fig. 7(a), Roberts is a fast response algorithm so ACR of S1 could be dropped very fast to 135 Mbps as explained in Fig. 6. However, Ren at Branch point B has to wait for the farthest BRM cell (from dS1) before it is able to send the BRM cell back to Branch point A. Hence, the time that S1 spends to drop its ACR to a fair share rate is 76 ms. This slow response is equal to the longest round trip time of the network.

2. Fig. 7(b) looks similar to Fig. 7(a) except for the temporary dropping to a fair share rate of ACR at 26 ms time. This is because Fahmy can detect an overloaded condition at Branch point B and immediately sending back a BRM



**Fig. 7.** ACR of the source S1 and S4 Roberts and others interoperation



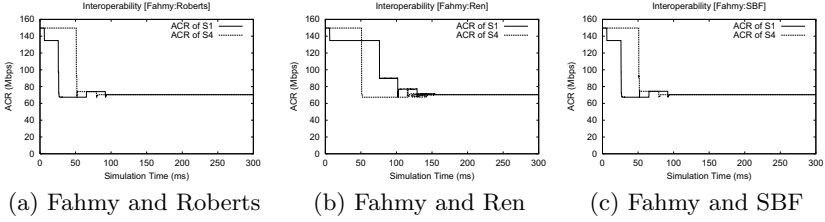
**Fig. 8.** ACR of the source S1 and S4 Ren and others interoperation

cell with 67.5 Mbps ER value to reduce the rate of source S1. During waiting for BRM from the farthest destination, Branch point B does not send any BRM cell. Hence, the next BRM cell received by S1 come from dS3 which has 135 Mbps ER value. Then the ACR goes abruptly and will caused a lot of queue build up at Branch point B.

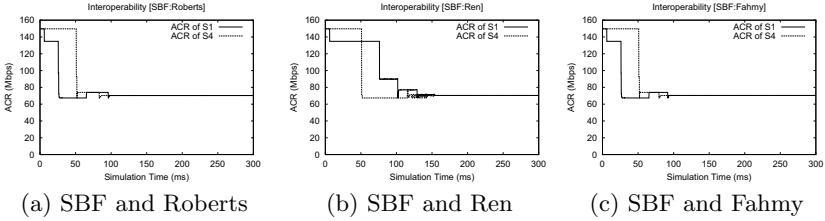
3. Roberts and SBF in Fig. 7(c) exhibits a similar response as in Fig. 7(b) during the 26 ms interval. After 26 ms, SBF at Branch point B still feedbacks the BRM with 67.5 Mbps from a selected branch to the source and the source send the data at this rate during 26 ms to 76 ms interval. However, the ACR oscillation still exists after the 76 ms. So far, we observe from Fig. 6 and Fig. 7 that if Branch point A is implemented with the algorithm that has fast response consolidation noise i.e. Robert, the ACR is oscillated no matter which algorithms are used at Branch point B.

4. Fig. 8(a) to Fig. 8(c) show the same pattern of ACR of S1 no matter what algorithm used at Branch point B. This is because Branch point A has to wait for all branches connecting to it before sending a BRM cell to S1. However, ACR remains at 150 Mbps until it drops to fair share (not drop to 135 Mbps before to fair share as in Fig. 7). Notice that in Fig. 8, ACR of S1 drops to fair share faster than that in Fig. 7(a). The reason is Roberts, Fahmy and SBF at Branch point B have a fast response so they can send a BRM to Branch point A within 26 ms. while Ren has to wait for BRM from dS1.

5. The results of using Fahmy and SBF at Branch point A are shown in Fig. 9 and Fig. 10. Fahmy and SBF overcome Roberts and Ren's drawbacks. The consolidation noise exists in Fig. 7 and the slow response exists in Fig. 8 are solved. However, if Ren is used at Branch point B, S1 will take longer time than using others before getting to a fair share rate. From Fig. 8 to Fig. 10 we notice that the response of the network trends to has the same characteristics



**Fig. 9.** ACR of the source S1 and S4 Fahmy and others interoperation



**Fig. 10.** ACR of the source S1 and S4 SBF and others interoperation

as the algorithm implemented at Branch point A. This inline with the first observation we made previously. Hence, we can conclude at this point that the most upper stream branch point dominates the response of the network while the lower stream ones have less effects. The simulation results of all cases of different consolidation algorithms used in the network are summarized in Table 2.

## 5 Conclusions

We have proposed a new efficient consolidation algorithm. The performance of the algorithm has been evaluated using a simulation. The results show that this new algorithm meets all features required in the point-to-multipoint ABR connections. It has a faster response, better link utilization and lower implementation complexity comparing to the previously proposed one. Especially, in the network that has a VBR as a background traffic. For the interoperability issue, a variety of the previously proposed consolidation algorithms have been interoperated. We can conclude that the most upper stream branch point is the most important one in the interoperation of consolidation algorithm in point-to-multipoint network. In order to avoid the consolidation noise and get a faster response, we recommend implementing a fast response and low consolidation noise algorithm such as Fahmy or SBF at the most upper stream branch point. We found that the network performance is not much affected by the behavior of the consolidation algorithms implemented at the lower stream branch points. Hence, the implementation complexity of the algorithm should be the major issue to be considered for using in these branch points.

**Table 2.** Simulation results for a network using different consolidation algorithm at Branch point A and Branch point B

Branch point A	Branch point B	Consolidation noise	Transient Response time* (ms)	Time to get to fair share rate (ms)
Roberts	Ren	Yes	6	-
	Fahmy	Yes	6	-
	SBF	Yes	6	26**
Ren	Roberts	No	26	26
	Fahmy	No	26	26
	SBF	No	26	26
Fahmy	Roberts	No	26	26
	Ren	No	26	$\geq 76$
	SBF	No	26	26
SBF	Roberts	No	26	26
	Ren	No	26	$\geq 76$
	Fahmy	No	26	26

\* The earliest time that S1 responses to the network.

\*\* Though consolidation noise exists, S1 gets to fair share rate during 26-76 ms period.

## References

1. The ATM Forum: The ATM forum traffic management specification version 4.0. (1996)
2. Ramakrishnan, K.K., Jain, R.: A Binary feedback scheme for congestion avoidance in computer networks. ACM Trans.Comp.Sys., Vol. 8. (1990) 158-181
3. Bonomi, F., Fendick, K.W.: The rate-based flow control framework for the available bit rate service. IEEE Network, (1995)
4. Tzeng, H.Y., Siu, K.Y.: On Max-min fair congestion control for multicast ABR service in ATM. IEEE Journal on Selected Areas in Communications, (1997)
5. Roberts, L.: Rate based algorithm for point to multipoint ABR service. ATM Forum 94-0772R1, (1994)
6. Ren, W., Siu, K.Y., Suzuki, H.: On the performance of congestion control algorithms for multicast ABR service in ATM. IEEE ATM'96 Workshop, (1996)
7. Cho, Y-Z., Lee, S-M., Lee, M-Y.: An Efficient Rate Algorithm for Point-to-Multipoint ABR Service. IEEE GLOBECOM, (1997)
8. Fahmy, S., Jain, R., Kalyanaraman, S., Goyal, R., Vandalore, B., Kota, S., Samudra, P.: Design and Evaluation of Feedback consolidation algorithm for ABR point-to-multipoint Connections in ATM Networks. IEEE INFOCOM'98, (1998)
9. Fahmy, S., Jain, R., Kalyanaraman, S., Goyal, R., Vandalore, B., Kota, S., Samudra, P.: Feedback consolidation algorithm for ABR point-to-multipoint connections. ATM Forum 97-0615, (1997)
10. Jiang, T., Zegura, E.W., Ammar, M.: Improved consolidation algorithms for point-to-multipoint ABR service. IEEE, 0-7803-4874-5/98. (1998)
11. Rangsinoppamas, N., Jaruvitayakovit, T., Sa-Niamsak, W., Prapinmongkolkarn, P.: Compromised Consolidation Algorithm for Point-to-Multipoint ABR Service in ATM Networks. ATCS'99, (1999)
12. Jain, R., Kalyanaraman, S., Goyal, R., Fahmy, S., Viswanathan, R.: ERICA switch algorithm: A complete description. ATM Forum 96-1172, (1996)

# Evaluation of the INSIGNIA Signaling System

Seoung-Bum Lee, Gahng-Seop Ahn, Xiaowei Zhang, and Andrew T. Campbell

COMET Group, Center for Telecommunication Research,  
Columbia University, New York, NY 10027  
insignia@comet.columbia.edu

**Abstract.** We present an evaluation of the INSIGNIA signaling system that supports quality of service in mobile ad hoc networks. INSIGNIA is based on an in-band signaling and soft-state resource management paradigm that is well suited to supporting mobility with end-to-end quality of service in highly dynamic environments where the network topology, node connectivity and radio channel conditions are time-varying. We evaluate the performance of the signaling system and show the benefit of our scheme under diverse mobility, traffic and channel conditions.

## 1 Introduction

Mobile ad hoc networks are autonomous distributed systems that comprise a number of mobile nodes connected by wireless links forming arbitrary time-varying wireless network topologies. Mobile nodes function as hosts and routers. As hosts, they represent source and destination nodes in the network while as routers, they represent intermediate nodes between a source and destination providing store-and-forward services to neighboring nodes. Nodes that constitute the wireless network infrastructure are free to move randomly and organize themselves in arbitrary fashions. Therefore the wireless topology that interconnects mobile hosts/routers can change rapidly in unpredictable ways or remain relatively static over long periods of time. These bandwidth constrained multi-hop networks typically support best effort voice and data communications where the achieved “goodput” is often lower than the maximum radio transmission rate after encountering the effects of multiple access, fading, noise, and interference, etc. In addition to being bandwidth constrained mobile ad hoc network are power-constrained because network nodes rely on battery power for energy. Providing suitable quality of service (QOS) support for the delivery of real-time audio, video and data in mobile ad hoc networks presents a number of significant technical challenges.

There has been little research in the area of supporting quality of service in mobile ad hoc networks. What work exists tends to be based on distributed scheduling algorithms [1] that address re-scheduling when the network topology changes and QOS-based medium access controllers. In [2] and [3], multi-hop multi-cluster packet radio network architectures are proposed. Provisioning quality of service is based on “dynamic virtual circuits” derived from network control and signaling found in ATM networks. This approach relies on a “circuit” model that requires explicit connection management and the establishment of “hard-state” in the network prior to communications. We believe there is a need to investigate alternative network models

other than those based on hard-state virtual circuits. New models need to be more responsive to the dynamics found in mobile ad hoc networks. Delivering end-to-end service quality in mobile ad hoc networks is intrinsically linked to the performance of the routing protocol because new routes or alternative routes between source-destination pairs need to be periodically computed during on-going sessions. The IETF Mobile Ad Hoc Networks (MANET) Working Group [4] recently began to standardize inter-network layer technologies (i.e., routing and support protocols). As such, it is presently focused on standardizing network-layer routing protocols suitable for supporting best effort packet delivery in IP-based networks.

In this paper, we present an evaluation of the INSIGNIA signaling system [5], which is a key component of a broader IP-based QOS framework [6] that supports adaptive services in mobile ad hoc networks. The signaling system plays an important role in establishing, adapting, restoring and terminating end-to-end reservations. Based on an in-band signaling approach, INSIGNIA supports fast reservation, restoration and adaptation algorithms. The structure of the paper is as follows. Section 2 presents an overview of the INSIGNIA signaling system. Following this, we present an evaluation of INSIGNIA in Section 3. We evaluate the suitability of the signaling system to support end-to-end service quality under various traffic, mobility and channel conditions. In Section 4, we make some final remarks.

## 2 Signaling System Overview

The INSIGNIA signaling system is designed to be lightweight in terms of the amount of bandwidth consumed for network control and to be capable of reacting to fast network dynamics such as rapid host mobility, wireless link degradation, intermittent session connectivity and end-to-end quality of service conditions. In what follows, we provide an overview of the signaling system. For full details of the INSIGNIA QOS framework and its signaling specification see [5] and [6], respectively.

### 2.1 In-Band Commands

Protocol commands support fast reservation, restoration and adaptation mechanisms. These commands are encoded using the IP option field and include *service mode*, *payload type*, *bandwidth indicator* and *bandwidth request* fields. By adopting an IP option in each IP packet header the complexity of supporting packet encapsulation inside the network is avoided.

When a source node wants to establish a fast reservation to a destination node it sets the reservation (RES) mode bit in the IP option service mode field of a data packet and sends the packet toward the destination. On reception of a RES packet, intermediate routing nodes execute admission control to accept or deny the request. When a node accepts a request, resources are committed and subsequent packets are scheduled accordingly. In contrast, if the reservation is denied packets are treated as best effort (BE) mode packets. In the case where a RES packet is received and no resources have been allocated the admission controller attempts to make a new reservation. This condition commonly occurs when flows are re-routed during the lifetime of an on-going session due to host mobility. When the destination receives a

RES packet it sends a QOS report to the source node indicating that an end-to-end reservation has been established.

The service mode indicates the level of service assurances requested in support of adaptive services. The interpretation of the service mode, which indicates a RES or BE packet, is dependent on the payload type and bandwidth indicator, respectively. A packet with the service mode set to RES and bandwidth indicator set to MAX or MIN is attempting to set-up a “max-reserved” or “min-reserved” service, respectively. The bandwidth requirements of a flow are carried in the bandwidth request field. A RES packet may be degraded to BE service in the case where re-routing or insufficient resources exist along the new/existing route. Note that a BE packet requires no resource reservation to be made. The IP option also carries an indication of the payload type, which identifies whether the packet is a base QOS (BQ) or enhanced QOS (EQ) packet. Using the ‘packet-state’ (service mode/payload type/bandwidth indicator) one can determine what component of the flow was degraded. Reception of a BE/EQ/MIN or RES/BQ/MIN packet indicates that the enhanced QOS packets have been degraded to best effort service. By monitoring the packet-state the destination node can issue scaling/drop commands to the source node.

## 2.2 Fast Reservation

To establish adaptive flows, source nodes initiate reservations by setting the appropriate field in the IP option in data messages before forwarding “reservation request” packets on toward destination nodes. A reservation request packet is characterized as having the service mode set to RES, payload set to BQ/EQ and bandwidth indicator to MAX/MIN and a valid bandwidth requirements. Reservation packets traverse intermediate nodes executing admission control modules, allocating resources and establishing flow-state at all intermediate nodes between source-destination pairs. A source node continues to send reservation packets until the destination node completes the reservation set-up phase by informing the source node of the status of the flow establishment phase using QOS reporting. When a reservation is received at the destination node, the signaling module checks the flow establishment status. The status of the reservation phase is determined by inspecting the IP option field service mode, which should be set to RES. If the bandwidth indication is set to MAX, this implies that all nodes between a source-destination pair have successfully allocated resources to meet the base and enhanced bandwidth needs in support of the max-reserved service mode. On the other hand, if the bandwidth indication is set to MIN this indicates that only the base QOS can be currently supported (i.e., min-reserved mode). In this case, all reservation packets with a payload of EQ received at the destination will have their service level flipped from RES to BE by the bottleneck node.

## 2.3 Soft-State Management

Reservations made at intermediate routing nodes between source and destination pairs are driven by soft-state management. A soft-state approach is well suited for management of resources in dynamic environments where the path and reservation associated with a flow may change. The transmission of data packets is strongly



coupled to maintenance of flow-states (i.e., reservations). In other words, as the route changes in the network, new reservations will be automatically restored by a restoration mechanism. Once admission control has accepted a request for a new flow, soft-state management starts a soft-state timer associated with the new or re-routed flow. The soft-state timer is continually refreshed as long as packets associated with a flow are periodically received at intermediate routers. In contrast, if packets are not received (e.g., due to re-routing) then the soft-state is not refreshed but times out with the result of deallocating any resources. Since data packets are used to maintain the state at intermediate nodes we couple the data rate of flows to the soft-state timer value. A major benefit of soft-state is that resources allocated during the reservation phase are automatically removed in a fully independent and distributed manner when the path changes.

## 2.4 Fast Restoration

Flows are often re-routed within the lifetime of on-going sessions due to host mobility. The goal of flow restoration is to re-establish reservation as quickly and efficiently as possible. Rerouting active flows involves the routing protocol [7] [8] [9] (to determine a new route), admission control and resources reservation for nodes that belong to a new path. Restoration procedures also call for the removal of old flow-state at nodes along the old paths. In an ideal case, the restoration of flows can be accomplished within the duration of a few consecutive packets given that an alternative route is cached. We call this type of restoration “immediate restoration”. If no alternative route is cached the performance of the restoration algorithm is coupled to the speed at which the routing protocols can discover a new path. When an adaptive flow is re-routed to a node where resources are unavailable, the flow is degraded to best effort service. Subsequently, downstream nodes receiving these degraded packets do not attempt to allocate resources or refresh the reservation state associated with the flow. In this instance the state associated with a flow is timed out and resources deallocated. A reservation may be restored if resources are free up at a “bottleneck” node or further re-routing allows the restoration to complete. We call this type of restoration “degraded restoration”. A flow may remain degraded for the duration of the session never restoring which we describe as “permanent degradation”. The enhanced QOS component of an adaptive flow may be degraded to best effort service (i.e., min-reserved mode) during the flow restoration process if the nodes along the new path can only support the minimum bandwidth requirement.

## 2.5 QOS Reporting

QOS reporting is used to inform source nodes of the on-going status of flows. Destination nodes actively monitor on-going flows inspecting status information (e.g., bandwidth indication) and measuring the delivered QOS (viz. packet loss, delay, throughput, etc.). QOS reports are also sent to source nodes to complete the reservation phase and on a periodic basis to manage end-to-end adaptation. QOS reports do not have to travel on the reverse path toward a source. Although QOS reports are generated periodically according to the applications’ sensitivity to the service quality, QOS reports are sent immediately when required (i.e., typically

actions related to adaptation). In the case where only BQ packets can be supported, as is the case with the min-reserved mode service, the signaling systems at the source “flips” the service mode of the EQ packets from RES to BE with all “degraded” packets sent as best effort. Partial reservations [6] that exist between source and destination nodes are automatically timed out after “flipping” the state variable in the EQ packets. Since there is a lack of EQ packets with the RES bit set at intermediate routers any associated resources are released allowing other competing flows to contend for these resources.

## 2.6 Adaptation

QOS reports are used as part of the on-going adaptation process that responds to resource changes in mobile ad hoc networks. Flow reception quality is monitored at the destination node and based on a user supplied policy, actions are taken to adapt flows under certain observed conditions. Action taken is conditional on what is programmed into the adaptation policy by the user. INSIGNIA provides a set of adaptation levels that can be selected. Typically, an adaptive flow operates with both its base and enhanced components being transported with resource reservation. Scaling flows down depends on the adaptation policy selected. Flows can be scaled down to their base QOS delivering enhanced QOS packets in a best-effort mode hence releasing any partial reservation that may exist. On the other hand, the destination can issue a drop command to the source to drop enhanced QOS packets (i.e., the source stops transmitting enhanced QOS packets). Further levels of scaling can force the base and enhanced QOS packets to be transported in best effort mode. In both cases, the time scale over which the adaptation actions occur is dependent on the application itself. These scaling actions could be instantaneous or based on a low-pass filter operation.

## 3. Evaluation

### 3.1 Simulation Environment

The INSIGNIA simulator consists of 19 ad hoc nodes. Each mobile node has a transmission range of 50 meters and shares a 2 Mbps air interface between neighboring mobile nodes within the transmission range. Time-varying wireless connectivity between nodes is modeled using 42 links. The mobility model is based on link failure and recovery characteristics; that is, connectivity is randomly removed and recovered with an arbitrary exponential distribution. Typically, mobile ad hoc networks do not have full connectivity between all mobile nodes at any given time due to the mobility behavior of mobile nodes and time-varying wireless link characteristics. With this in mind, maximum network connectivity is set at 85% such that 15% of the mobile nodes within their transmission ranges remain disconnected.

The architectural [6] components implemented in our simulator include: (i) the signaling system; (ii) the Temporally Ordered Routing Algorithm (TORA) [7]; (ii) a packet scheduler based on a deficit round robin implementation; and (iii) an admission controller, which is simply based on peak rate bandwidth allocation.

For simulation purposes 10 adaptive flows with different bandwidth requirements ranging from 75-500 kbps are operational throughout the simulation. An arbitrary number of best effort flows are randomly generated to introduce different loading conditions distributed randomly throughout the network (i.e., in different parts of the network) during the simulation. We also chose an arbitrary traffic pattern/load with average packet size of 2 Kbytes. Identical traffic/load is used for all scenarios under investigation. The base QOS component of adaptive flows corresponds to 50-70% of an adaptive flow's bandwidth needs whereas enhanced QOS corresponds to 30-50%. For example, an adaptive flow of 300 kbps may have a base and enhanced QOS of 150 kbps for each component such that minimum and maximum bandwidth requirement is set to 150 and 300 kbps, respectively.

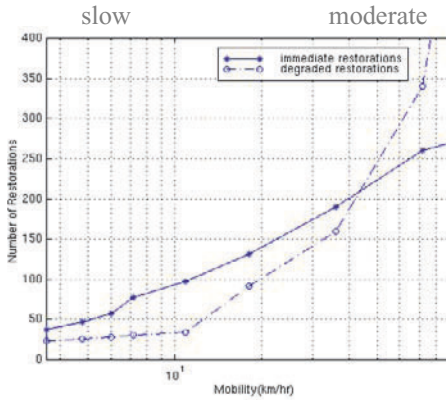
The mobility model used throughout the simulation supports three different rates of mobility. Moderate mobility represents slow vehicular mobility ranging from 9-18 km/hr. Mobility conditions slower than moderate mobility is defined as slow mobility (i.e., speed less than 9km/hr) while rates faster than moderate mobility are categorized as fast mobility (i.e., speed exceeding 18km/hr). We inherit the mobility model used in the TORA simulation [10].

For the purpose of evaluating our signaling scheme, we measure per-session and aggregate network conditions for a number of experiments and analyze flow restoration, soft-state management and host/router mobility. We observe throughput, delays, out-of-order sequence packets, lost packets, percentage of delivered degraded packets for the different mobility rates and system configurations (e.g., changing soft-state timers). We are particularly interested in percentage of reserved and degraded packets delivered to receivers. This metric represents the ability of our framework to deliver assurance. We also observe the number of re-routing, degradation, restoration and adaptation events that took place during the course of each experiment as a measure of the dynamics of the system under evaluation.

### 3.2 Restoration Analysis

In the following experiment we investigate the impact of re-routing and restoration on adaptive flows. Since re-routing of flows requires admission control, resource allocation, state creation and removal of old state, we track the re-routing and restorations events and any degradation that takes place.

Typically, adaptive flows experience continuous re-routing during their session holding times. This is certainly the case with flows that represent continuous audio and video flows but not necessarily the case for micro-flows. Flows may be re-routed over new paths that have insufficient resources to maintain the required QOS. A key challenge for restoration is the speed at which flows can be restored. This is dependent on the speed at which new routes can be computed by the routing protocol if no alternative routes are cached and the speed at which the signaling system can restore reservations. The speed at which old reservations are removed is a direct function of the soft-state timer. The mobility rate impacts the number of restorations observed in the system and therefore the QOS delivered by INSIGNIA. As the rate of mobility increases (e.g., from moderate to fast mobility) restoration algorithms need to be scalable and highly responsive to such dynamics in order to maintain end-to-end QOS.



“best effort to min-reserved” (meaning that the min-reserved flow is degraded to best effort before being restored to min-reserved) only occur when the re-routed adaptive flows encounter resources to support min-reserved service. We observed that degraded restoration for “best effort to max-reserved” is the most dominant degraded restoration type observed as shown in Figure 2. This is because re-routed flows are more likely to be accepted or denied rather than degraded to min-reserved flows under slow and moderate mobility conditions. However, we observe that when the mobility is fast that “best effort to min-reserved” degraded restoration becomes the dominant type as shown in Figure 2. In the case of high mobility, only a limited number of routes exist to route flows causing service degradation. The rapid fluctuations in the monitored QOS causes the adaptation process at the destination to request that the degraded flows be scaled down to their min-reserved mode. In this instance, the “best effort to min-reserved” restoration becomes the dominant type as shown in Figure 2.

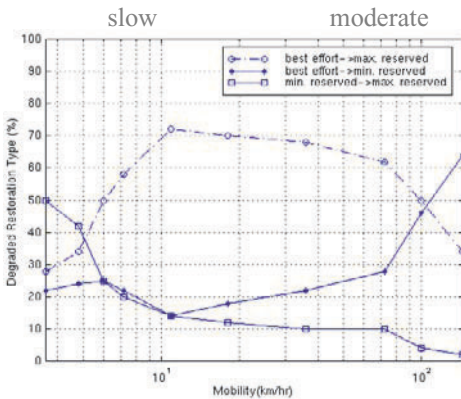


Fig. 2. Degradeestorations Types

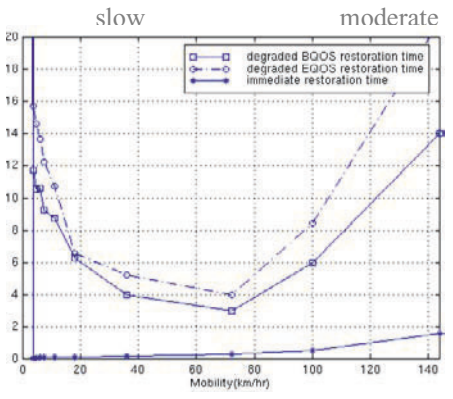


Fig. 3. Time Spent for Immediate Restorations and Degraded Restorations

Increased mobility forces mobile hosts to adapt flows to their min-reserved modes preventing adaptive flows from scaling back up due to the fast time scale dynamics and re-routing experienced. When the mobility is fast, all adaptive flows scale down to their min-reserved service 90 seconds into the simulation trace but only two scale-up adaptations actions are observed during the complete trace. The number of “best effort to max-reserved” and “min-reserved to max-reserved” degraded restorations decreases as mobility is increased to fast as shown in Figure 2. The “best effort to min-reserved” degraded restorations continue to increase implying that most flows scale down to their minimum requirements and operate at the min-reserved mode. Figure 3 illustrates the restoration times across the complete mobility range. The base QOS restoration time corresponds to the time taken to regain the min-reserved service for a flow that has been temporarily degraded to best effort mode. The enhanced QOS restoration time corresponds to the time taken for the max-reserved service to restore from best effort service or from min-reserved service. We observe that the average restoration time required for immediate restoration is relatively constant at 0.2 ~ 0.9 sec under all mobility conditions. In this case immediate restorations only require an interval of two consecutive packets to restore a reservation. However, unlike the

immediate restorations mobility conditions impact the average degraded restoration times as shown in Figure 3.

### 3.3 Soft-State Analysis

Mobile soft-state management is used to maintain reservations. The duration of soft-state timers has a major impact on the utilization of the network. Figure 4 shows the impact of soft-state times on network performance in terms of the number of reserved mode packets delivered. Reception of a reserved mode packet (with the service mode set to RES) at the destination indicates that the packet is delivered with max-reserved or min-reserved assurance. Reception of a degraded packet implies that the packet has been delivered without such guarantees. Therefore the percentages of reserved and degraded packets received by destination nodes as a whole indicate the degree of service assurance that an INSIGNIA network can support for different values of soft-state timers.

In what follows, we discuss the impact of mobile soft-state timers on network performance. We set the mobile soft-state timer value in the range of 0.01 to 30 sec and observe the corresponding system performance. For each experiment we set the same timer value at each node. As shown in Figure 4, the soft-state timer value has an impact on the overall network performance. The ability to support adaptive services decreases as the soft-state timer value increases. The percentage of delivered reserved packets decreases as mobile soft-state timer increases. The percentage of degraded packets increases as the soft-state timer value increases as shown in Figure 15. Worst case performance is observed when the soft-state timer value is set to 30 sec. In contrast, the best performance is observed when soft-state timer is set to 2 sec as shown in Figure 4. We observed that 69% of the packets are delivered as reserved packets and 31% as best effort packets when the soft-state timer is set to 30 sec. Support for QOS substantially improves with 88% of reserved packets being delivered to receivers with a soft-state timer value of 2 sec. Large timeout values tend to lead to under utilization of the network because resources are “locked up” where resources remain allocated long after flows have been re-routed. New flows are unable to use these dormant resources resulting in the overall degradation of the network due to “resource lock-up”.

As the value of the soft-state timer gets smaller fewer resource lock ups are observed and utilization increases. However, when the timer is set to a value smaller than 2 sec the network experiences what we describe as “false restoration”. This occurs when a reservation is prematurely removed because of a small soft-state timer. However, this is a false state because the session holding time is still active and the source node keeps sending packets. In this case, the reservation is removed because of a timeout and then immediately reinstated when the next reserved packet arrives. False restorations occur when the timeout value is smaller than the inter-arrival time between two consecutive packets associated with a flow. With a soft-state timer of 0.04 sec, for example, all the adaptive flows experienced numerous false restorations. Mobile routers often de-allocate and reallocate resources without the involvement of any network dynamics due to mobility. In the worst case, every packet can experience a false restoration. Such events not only increase the processing costs of state creation and removal, and resource allocation and de-allocation but also falsely reflect the resource utilization and availability of the system. When the network experiences

numerous false restorations, re-routed flows often find nodes with few resources allocated on the new path. This phenomenon causes flows to always gain max-reserved mode resources with mobile nodes accepting the request for resources well beyond their actual capacity. This results in reserved packets experiencing indefinite delays at intermediate nodes even though resource assurances are provided by admission control resulting in wide scale packet losses and service degradation. Figure 4 shows a “false restoration region” where there is little distinction between reserved and best effort operational modes and where reservations are typically always granted. Adaptation and restoration algorithms can fail under false restoration conditions due to perception of unlimited resource availability. Setting a suitable soft-state timer value is therefore essential to preventing both false restoration and resource lock-up in our framework.

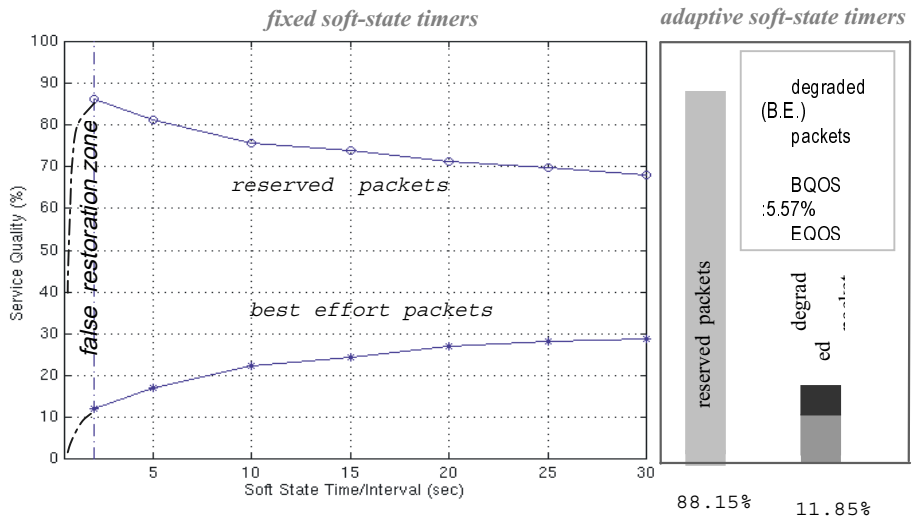


Fig. 4. Soft-State Timers and Network Performance

Each data packet associated with a reserved flow is used to refresh soft-state reservations. We observe that different adaptive flows have different data rates thus a fixed timeout value is too limiting. For example, one value may be fine for some set of flows but cause false restorations or resource lock-up for others. Clearly there needs to be a methodology for determining the value of the soft-state timer. The issue of false restoration and resource lock-up can be only resolved by adjusting the timeout values based on measured flow dynamics. The timeout should be based on the effective data rate of each flow. More specifically, the soft-state timer should be based on the measured packet inter-arrival rate of adaptive flows. The signaling system measures packet inter-arrivals and jitter at each mobile node for each flow, adjusting the soft-state timeout accordingly. In the experimental system we implemented an adaptive soft-state timer that is initially set to 4 sec representing an initial safety factor. This allows mobile nodes to set their soft-state timers according to their effective data rate allowing the timeout to adjust to network dynamics and the variation in the inter-arrival rates of individual flows traversing nodes. The implementation of an adaptive soft-state timeout effectively removes resource lock-

ups and false-restorations as shown in Figure 4. We observe that when an adaptive soft-state timer scheme is adopted 86% of flows are delivered as reserved packets, 11% as degraded packets and 3% of packets are lost. Adaptive soft-state timers greatly reduce resource lock up and false restoration conditions allowing the network to support better service assurances through the delivery of more reserved packets and fewer degraded packets at destination nodes.

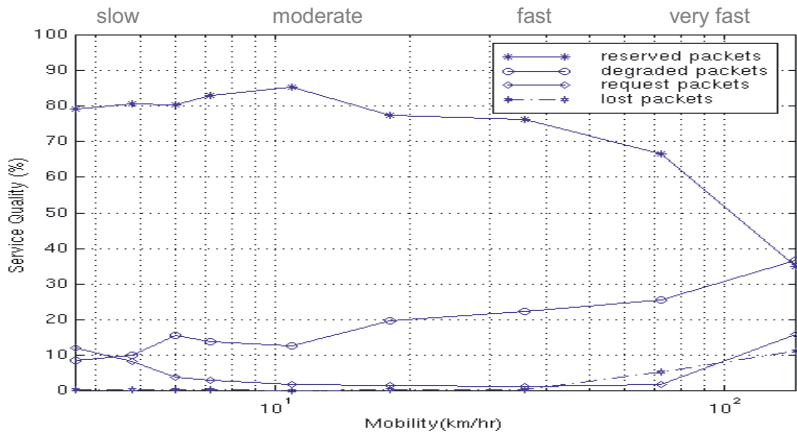


Fig. 5. Mobility and Network Performance

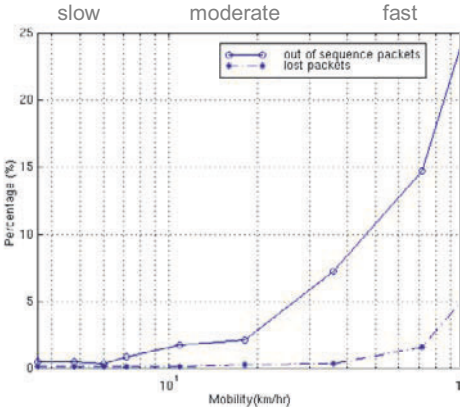
### 3.4 Mobility Analysis

To evaluate the impact of mobility on the INSIGNIA, we conduct a set of experiments operating under identical traffic pattern/load conditions and various mobility conditions ranging from 0 km/hr to 72 km/hr. Figure 5 shows the impact of mobility on the delivered service quality. When there is no host mobility, results closely approximates a fixed network infrastructure where admitted flows receive stable QOS assurances. One anomaly is observed, however. Six adaptive flows failed to be granted reservations due to lack of network resources at intermediate nodes. As a consequence only 49% of the packets are delivered as reserved packets and 51% as best effort packets. This anomaly is a product of the routing protocol, which provides a non-QOS routing solution. Adaptive flows are routed to bottleneck nodes resulting in the failure of admission control due to the lack of resources. This problem could be resolved by designing a signaling system that takes alternative routes in the case that admission control fails along a selected path.

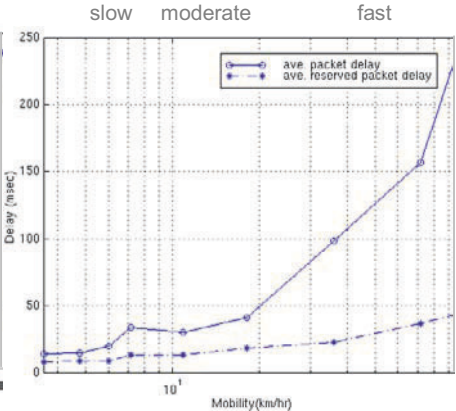
With the introduction of mobility into the network, the performance improves (i.e., more reserved packets are delivered) as illustrated in Figure 5. Mobility induced re-routing allows request packets to traverse alternative paths increasing the probability of finding a route with sufficient resource availability to admitted flows as reserved mode packets. Figure 5 shows that the INSIGNIA supports relatively constant QOS under slow and moderate mobility conditions between 3.6-18 km/hr. The optimal performance is observed when the average network mobility is approximately 11 km/hr. This results in the delivery of 86% of reserved packets. The



in-band nature of INSIGNIA allows the system to cope with fast network dynamics in a responsive manner. In an ideal case, INSIGNIA requires only a single packet reception to set up and restore (i.e., immediate restorations) reservation for the new or re-routed flows, respectively. INSIGNIA supports the delivery of 66% reserved packets even when mobile hosts are moving at 72 km/hr as shown in Figure 5. This is a very encouraging result.



**Fig. 6.** Impact of Mobility on Out of Order Delivery and Packet Loss



**Fig. 7.** Packet Delays

Note that the service provided in a mobile ad hoc network has a memoryless property such that adaptive flows require new admission tests along the new path when re-routing occurs. This implies that an increase in mobility may cause fluctuations in perceived service quality. At 72 km/hr all flows are scaled-down to min-reserved packets (after 90 sec into the simulation) due to fluctuations in the delivered quality. At this speed only two flows are capable of regaining their max-reserved service. When mobility conditions exceed 72 km/hr support for QOS breaks down rapidly as indicated in Figure 5. The mobility characteristics overload the system and service assurance for adaptive flows diminish. In fact, when mobility exceeds 90 km/hr, we observe that a number of flows are transported as best effort packets for more than 70 sec because they failed to accomplish their end-to-end flow set up due to persistent loss of RES packets and QOS reports. This phenomenon corresponds to the abrupt loss of reserved packets and degraded packets.

An increase in out-of-sequence packet is also observed at higher speeds possibly causing service disruption at the receiver. Figure 6 shows the number of out-of-sequence packets under various mobility conditions. The number of out-of-sequence packets generally increases as mobility increases. The different propagation delay characteristics of reserved and best effort packets associated with the same end-to-end flow has an impact on out-of-sequence packets. Figure 6 also shows the number of lost packets observed under different mobility conditions. Packets that are delayed for more than 15 sec are discarded at intermediate nodes and are considered to be lost packets. Figure 7 shows the delay characteristics of packets under various mobility conditions. When mobility increases, the connectivity between nodes becomes problematic. Such network dynamics trigger frequent routing updates and decreased

connectivity. Thus, the number of available routes between nodes decreases as mobility increases. Degraded packets queue up at intermediate nodes experiencing long delays. However, the reserved packets are less sensitive to these delays as indicated in Figure 7 with all reserved packets being delivered within a period of 40 msec.

## 4 Conclusion

In this paper, we have presented the evaluation of the INSIGNIA signaling system, an in-band signaling system that supports fast reservation, restoration and adaptation algorithms that are specifically designed to deliver high performance adaptive services in mobile ad hoc networks. The signaling system is designed to be lightweight and highly responsive to changes in network topology, node connectivity and end-to-end quality of service conditions. Our simulation results show the benefit of INSIGNIA under diverse mobility, traffic and channel conditions in support of fast reservation, restoration and adaptation. The use of in-band signaling and soft-state resource management proved to be very efficient, robust and scalable under a wide variety of network conditions. Our results highlighted a number of anomalies that emerged during the evaluation phase. However, the use of adaptive soft-state timers resolved these issues.

## Acknowledgements

This work is supported in part by the Army Research Office under award DAAD19-99-1-0287. We are particularly indebted to Vincent Park and Dr. M. Scott Corson for providing the INSIGNIA project with the source code for TORA.

## References

1. Ephremides, A. and T. Truong, Scheduling Algorithm for Multi-hop Radio Networks, *IEEE Trans. on Computers*, 38:1353, 1989.
2. C. R. Lin and M. Gerla, Asynchronous Multimedia Multihop Wireless Networks, in *Proc. IEEE INFOCOM'97*, April 1997.
3. R. Ramanathan and M. Streenstrup, Hierarchically-Organized, Multi-hop Mobile Wireless networks for Quality-of-service Support, Technical Report, BBN, 1998.
4. J. Macker, and M. S. Corson, Mobile Ad hoc Networking (MANET): Routing Protocol Performance Issues and Evaluation Considerations, draft-ietf-manet-issues-01.txt, *Internet Draft*, work in progress, April 1998.
5. Lee, S-B and A. T. Campbell, INSIGNIA, *Internet Draft*, draft-lee-insignia-00.txt, work in progress, November 1998.
6. Lee, S.B., Gahng-Seop, A., Zhang, X., and A.T. Campbell, INSIGNIA: An IP-Based Quality of Service Framework for Mobile Ad Hoc Networks, *Journal of Parallel and Distributed Computing* (Academic Press), April 2000.

7. V. Park, and S. Corson, Temporally Ordered Routing Algorithm (TORA) Version 1 Functional Specification, draft-ietf-manet-tora-spec-00.txt, *Internet Draft*, work in progress, November 1997.
8. C. Perkins, Ad hoc On demand Distance Vector (AODV) Routing, draft-ietf-manet-aodv-01.txt, *Internet Draft*, work in progress.
9. D. B. Johnson and D. A. Maltz, Dynamic Source Routing in Ad Hoc Wireless Network, *Mobile Computing*, Chapter 5, pp. 153-181.
10. TORA OPNET source code supplied by V. Park and M. S. Corson, 1997.

# Design and Implementation of RSVP Based on Object-Relationships

Martin Karsten<sup>1</sup>

Industrial Process and System Communications  
Darmstadt University of Technology  
Merckstr. 25 • 64283 Darmstadt • Germany  
phone: +49-6151-166156 • fax: +49-6151-166152  
email: [Martin.Karsten@KOM.tu-darmstadt.de](mailto:Martin.Karsten@KOM.tu-darmstadt.de)  
<http://www.kom.e-technik.tu-darmstadt.de>

**Abstract.** RSVP has been proposed by the IETF as a signalling protocol for reservation-based quality-of-service enabled communication in IP networks. While RSVP's concepts are very sophisticated, further research efforts and potential modifications might be necessary to accomplish additional requirements before general deployment and commercial usage. Currently, only one freely available implementation exists and even some of the commercial implementations are based on it. In this paper, an alternative approach to describe RSVP protocol operations is presented, employing relational specification of state blocks and object-relationships between them. The result appears to be more concise and comprehensible than existing processing rules, yet not giving up efficiency. An implementation design based on this methodology, as well as specific details and optimizations are derived and explained. The implementation is designed to be portable across different operating system platforms and even to simulation environments. The primary purpose is to carry out research on modifications of RSVP, being able to examine those by simulation, emulation and real tests. Applying these considerations, an experimental protocol engine has been implemented, which is publicly available.

## 1 Introduction

RSVP (Resource ReSerVation Protocol), initially designed and described in [1], has been specified by the IETF [2] to carry reservation requests for communication resources across IP networks. Because RSVP is designed to handle requests for arbitrary service classes, an even more general point of view can be adopted by regarding it as a universal signalling protocol to carry quality of service requests.

For a variety of reasons [3], I believe that further research is needed to determine the best design of such a signalling protocol. This research can be grounded on the existing specification of RSVP, because of both its basic existence and its

---

<sup>1</sup>This work is sponsored in part by: Volkswagen-Stiftung, Hannover, Germany.

sophisticated design. The only existing freely available implementation [4] is not considered well-suited for such research (see [3] for details). An attempt to create a more suitable test environment should adhere to the following design objectives:

- structured (object-oriented) design and implementation
- portability for multiple platforms, including simulators
- clear representation of RSVP's concepts in the code

While at a first glance RSVP seems to be straightforward and easy to understand, the details of an implementation are rather complex. The goals of this project are twofold. The first goal is to specify protocol operations more comprehensible than existing documentation does. The second goal is to create and publish an experimental platform which allows researchers to test and examine modifications with reasonable effort. The context and initial motivation of this implementation project are in the areas of charging for QoS in network communication [5,6,7] and interoperability of heterogeneous QoS architectures [8,9].

The rest of this paper is organized as follows. In the next section, a brief overview of RSVP is given, adopting the terminology of [2]. A specification of RSVP message processing, based on object-relationships, is presented in Section 3. In Section 4, an appropriate software design approach is derived from this specification. The current status of the implementation with respect to the RSVP specification is described in Section 5. Section 6 concludes the paper with a summary and an outlook on further work items. Note that this paper is shortened in order to accomplish the space limitation for its publication here. A more detailed version is available as [3].

## 2 RSVP Overview

RSVP is designed to carry reservation requests for packet-based, stateless network protocols such as IP (Internet Protocol). In essence, it is aimed at combining the robustness of connectionless network technology with flow-based reservations by following a so-called *soft state* approach. State is created to manage routing information and reservation requests, but it times out automatically, if it is not refreshed periodically. In the RSVP model, senders inform RSVP-capable routers and receivers about the possibility for reservation-based communication by advertising their services via PATH messages. These messages carry the sender's *traffic specification* (TSpec) and follow exactly the same path towards receivers as data packets, establishing soft state in routers. Receivers initiate reservations by replying with RESV messages. They contain a TSpec and a *reservation specification* (RSpec) and also establish soft state representing the reservation. RESV messages are transmitted hop-by-hop and follow exactly the reverse path that is formed by PATH messages.

RSVP treats reservation requests (e.g. TSpec and RSpec) as opaque data and hands them to complementary local modules, which are able to process them appropriately. Being tuned to support large multicast groups, RSVP uses logic from these modules to merge reservation requests that share parts of the transmission path. Merging takes place at outgoing interfaces by merging requests from different next hops that can be satisfied by a single reservation at the same interface. As well,

reservation requests that are transmitted towards a common previous hop are candidates for merging. The amount of merging possible is determined by the filter style, which is requested by receivers. For shared filter style, all reservations for the same interface and all reservations towards the same previous hops are merged, respectively. When distinct filter style is requested, only reservations that specify the same sender are being merged. Furthermore, filter styles are classified by whether applicable senders are wildcarded or listed explicitly. The (potentially empty) list of senders is called *FilterSpec*. The following filter styles are currently defined:

- FF (fixed filter): single sender, distinct reservation
- SE (shared explicit): multiple senders, shared reservation
- WF (wildcard filter): all senders, shared reservation

All these filter styles are mutually exclusive and a session's filter style is determined from the first arriving RESV message. The combination of TSpec and RSpec is called *flow specification* (FlowSpec). The combination of FlowSpec and FilterSpec is referred to as *flow descriptor*.

### 3 Specification of RSVP Message Processing

In this section, a specification of RSVP message processing is presented, based on relational design and object-relationships between state blocks. A rigorous approach for modelling RSVP would begin by representing state information as relations and identifying functional dependencies between them. Then, well-known normalisation algorithms could be applied to create the highest possible normal form and message processing could be expressed using relational algebra. Intuitively, this is often done to some extent by software designers and programmers.

In this work, while not following the strict method, state information is explicitly modelled as relations which in turn are considered as state blocks to create object-relationships between them. The initial relational model is deduced from the relevant standardization documents [2,10,11] and personal reasoning about the protocol. Additionally, experiences made during design and implementation of the software have been a source of insight into protocol operations. We omit the details of relational representation here for reasons of brevity and refer to [3].

A significant part of RSVP message processing consists of finding appropriate state blocks for certain operations. For normal implementation (i.e. without using a relational database), state blocks and object-relationships are considered to be more expressive and efficient than directly implementing the relational model. The relationships between objects are explicitly stored when knowledge is available, instead of recalculating them through relational rules whenever they are needed. The algorithmic description in [10,11] exhibits a relational style, but without being rigorous. Opposite to that approach, the processing rules in this paper are based on object-relationships between state blocks. A subset of state blocks is similar to those described in [10,11], but semantics and lifetime are occasionally modified. Additional relations are designed to express useful state information. Eventually, these are represented as state block objects as well, to efficiently accomplish certain operations.

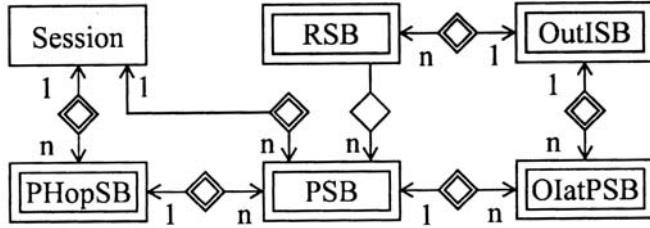


Fig. 1. Entity-Relationship Diagram for State Blocks

### 3.1 State Blocks and Relationships

From the initial relations, corresponding state block objects and state block relationships are deduced. Although this is not done rigorously (i.e. by using normalisation algorithms), certain optimizations are possible to avoid redundancy of information and suit an efficient implementation. The result can be expressed as an Entity-Relationship Model (ER-Model) and is shown as diagram in Figure 1.

#### 3.1.1 State Blocks

**Session.** For each RSVP session, the Session state block bundles all relevant information and the session's destination address and port is saved there. Relationships are kept to those state blocks that are needed to fully access all information. All Session objects are bound to a single RSVP object, representing an RSVP router.

**Path State Block (PSB).** A PSB holds all relevant information from a PATH message, i.e., the sender's address and traffic specification, routing information, etc.

**Reservation State Block (RSB).** An RSB represents a reservation requested from a next hop, particularly by holding the reservation specification, i.e., the FlowSpec, which determines the amount of resources that are requested, depending on the service class. It identified by its owning OutISB and the next hop's address.

**Outgoing Interface State Block (OutISB).** This state block represents the merged reservations from multiple RSBs applying at a certain outgoing interface. It is roughly comparable to the TCSB in [10,11]. However, different from those processing rules, the TCSB is split into a general (OutISB) and a specific part. The nature of the specific part depends on the particular traffic control implementation, which in turn depends on the corresponding link layer medium behind the interface [2,12,8]. An instance of OutISB is constructed immediately upon creation of the first contributing RSB.

**Outgoing Interface at PSB (OlatPSB).** For each outgoing interface that is part of the routing result for a PATH message, an instance of OlatPSB is created. A relationship to an OutISB object expresses that an actual reservation is active at this interface. The introduction of this state block allows to split the N:M relationship between PSB and OutISB into two 1:N relationships, which simplifies implementation.

**Previous Hop State Block (PHopSB).** The concept of an explicit PHopSB is new to an RSVP description. It is used to hold information about reservations that are merged at a certain incoming interface towards a previous hop, as well as the resulting

reservation request that is sent to this hop. A PHopSB is identified by the previous hop's IP address and the incoming interface, at which traffic from this hop arrives for the destination address of a session. Again, an object is created as soon as the first PATH message arrives from a certain previous hop.

In the rest of the paper, the terms *state block* and *state block object* are used synonymously. In Figure 1, all entities except Session are weak entities, i.e., they cannot uniquely be identified without the respective session key. Furthermore, OutISB is indirectly identified through an OIatPSB at any of the PSBs it applies to, although the cardinality ratio implies the opposite direction. In RSB there is no information about the outgoing interface stored and the list of senders is not used for identification. Thereby, it is a weak entity depending on the key of OutISB. For both RSB and OutISB, instead of storing the set of applicable senders, a relationship to PSB is maintained (indirectly in case of OutISB). The cardinality ratio of each relationship is shown in the diagram in Figure 1.

### 3.1.2 Relationships

Each relationship is presented, including the necessary key to traverse it, if it is a multi-object relationship. The respective keys applying to these relationships are often smaller than the full key of each state block. This is due to inherent identification through the relationships. However, the implementation of this model is done by directly storing the relationships. Furthermore, all Session objects are bundled into a global container. This could be considered as a special relationship to a unique object representing the RSVP router.

**SessionPSB**  $\leftarrow \diamond \rightarrow_n$  (1)key for PSB: Sender, Incoming Interface and Previous Hop  
In a PSB object, information about incoming interface and previous hop are not stored directly, instead this information can be extracted from the corresponding PHopSB (see Relationship (7) below).

**SessionPHopSB**  $\leftarrow \diamond \rightarrow_n$  key for PHopSB: Incoming Interface and Previous Hop (2)

**OutISBRSB**  $\leftarrow \diamond \rightarrow_n$  key for RSB: Next Hop (3)

Each OutISB is related to those RSBs that are merged together at an outgoing interface. Because an RSB only contributes to one specific OutISB, partitioning the set of RSBs along their OutISBs creates a complete and disjunct decomposition of all RSBs. Therefore, a relation between Session and RSB is not necessary to access RSBs from a Session object.

**RSBPSB**  $\rightarrow \diamond \rightarrow_n$  key for PSB: Sender (4)

A reservation applies to a set of senders, either by explicit selection (SE or FF filter style) or implicit association (WF filter style). Instead of storing a list of all sender addresses, a relationship to the respective PSBs is maintained from the RSB.


**OIatPSBPSB**  $\leftarrow \diamond \rightarrow_n$  key for OIatPSB: Outgoing Interface (5)

A merged reservation, installed at an outgoing interface, applies to a set of senders. As with RSBs, this is expressed by storing relationships to the respective OIatPSBs, instead of their address/port pairs.

**OIatPSBOutISB**  $\leftarrow \diamond \rightarrow_n$  key for OIatPSB: Sender (6)

This relationship expresses the reservation at a certain outgoing interface that is applied to traffic from a sender.



**PHopSBPSB**  key for PSB: Sender (7)

Each PSB is logically connected to the PHopSB representing its previous hop. This relationship is mainly used when reservation requests are created for previous hops. Information from the PHopSB (hop address and incoming interface) is used to distinguish PSB objects (see Relationship (1) and Path State relation).

## 3.2 Operations

In this section, the core operations of the RSVP protocol engine are explained with respect to the relationships between state blocks. The presentation is divided into 4 parts, which together form the central RSVP operations:

- State Maintenance
- Outgoing Interface Merging
- Incoming Interface Merging
- Timeout Processing

In general, if a message or timeout triggers a modification of internal state, all relationships are updated immediately during State Maintenance or Timeout Processing, except the relationship between PSB and OutISB. For Outgoing Interface Merging, the “old” state of this relationship has to be available to appropriately modify the filter setting at the underlying traffic control module. Afterwards, this relationship is updated, as well. If the contents of an RSB change, Outgoing Interface Merging is invoked. If during Outgoing Interface Merging, the contents of an OutISB are changed, Incoming Interface Merging is triggered. Only if the resulting reservation request (stored in PHopSB) changes, a new RESV message is created and sent to the previous hop.

The basic claim of this work is that maintaining relationships imposes no significant additional overhead during analysing an incoming message and updating state from it. However, when it comes to merging reservations and timeout processing, existing relationships can be used instead of recomputing them every time, especially under stable conditions. In this section, only the basic operations are described. Whenever the term *interface* is used in the following subsections, it might also denote the API (application programming interface). Again, additional details about message pre- and post-processing can be found in [2,10,11].

### 3.2.1 State Maintenance

Arriving RSVP messages are decomposed into components and processed depending on the type of message. During processing, appropriate state blocks have to be located, created and/or modified. In the following, a pseudo-algorithmic description of the processing rules are given for each message type. Although it is not mentioned explicitly for most of the message types, usually the appropriate Session object has to be determined first.

**PATH.** Find a Session and check for conflicting destination ports. If no Session exists, create one. Find a PSB for this sender through Relationship (1) and check for conflicting source ports. If none exists, create a new PSB. When creating a PSB object, create the relationship to the corresponding Session object. If the PSB is new

and no appropriate PHopSB can be found, create a new PHopSB. Set a relationship between PSB and PHopSB. If the session address is multicast and the incoming interface differs from the routing lookup result, mark this PSB as local to an API session. Update all information in the PSB and in case of relevant changes, trigger an immediate generation of a PATH message and potentially invoke Outgoing Interface Merging (Section 3.2.2).

**RESV.** Process each flow descriptor separately, i.e., each pair of FilterSpec and FlowSpec. Find or create an appropriate OutISB through Relationship (5) and (6) and find or create an RSB using Relationship (3). When creating new objects, set the corresponding relationships. Match (i.e. consider the intersection) the filter specification (in case of FF or SE) or the address list determining the scope (WF) against all existing PSBs that route to the outgoing interface through Relationship (1). Update the RSB and invoke Outgoing Interface Merging, if relevant content has changed, e.g., FilterSpec or FlowSpec.

**PTEAR.** Find a PSB through Relationship (1). If found, forward the message to the PSB's outgoing interfaces, remove the PSB, clear its relationships and invoke Outgoing Interface Merging.

**RTEAR.** Process each flow descriptor separately. Find an RSB through Relationship (5) and (6) and Relationship (3). If found, remove the filters that are listed in the message and invoke Outgoing Interface Merging. If the RSB's filter list is empty, remove the RSB and clear its relationship to OutISB.

**PERR.** Find a PSB through Relationship (1). If found, forward the message through the PSB's incoming interface.

**RERR.** Find a PHopSB for the previous hop address from the message. If found and the error code indicates an admission control failure, set a blockade FlowSpec at those PSBs from the PHopSB that match a filter from the message. Find all OutISBs that match a filter from the message and do not belong to the incoming interface. Forward the message to all RSBs that have a relationship to these OutISBs. In case of admission control failure, forward the message to only those RSBs that do not have a FlowSpec strictly smaller than that of the message.

**RCONF.** Forward the message to the outgoing interface that results from a routing lookup for the message's destination address.

### 3.2.2 Outgoing Interface Merging

During the merge operation at an outgoing interface, all applicable PSBs and RSBs have to be collected to access their TSpecs and FlowSpecs. Precise operation depends on the nature of the underlying link layer and appropriate algorithmic descriptions can be found for point-to-point or broadcast media in [10,11] and for non-broadcast multi-access media (e.g. ATM) in [13,12,8]. Outgoing Interface Merging operates on a certain OutISB. Relationships to those PSBs that are relevant and route to this interface as well as RSBs that contribute to the merged reservation state are known and can be traversed directly, instead of recomputing them. Therefore, no special (filter style dependent) rules have to be given on how to find those state blocks, but instead only rules to process them appropriately are necessary. The result is stored in the OutISB and, if the merged FlowSpec or the FilterSpec has changed, the appropriate PSBs and

PHopSBs (accessible through Relationship (5), (6) and (7)) are marked for Incoming Interface Merging. Certain policing flags have to be passed to traffic control, which can be derived from accessible information, as well. To determine whether this reservation is merged with any other reservation that is not less or equal, the LUB (least upper bound) of all merged FlowSpecs from all OutISBs (at different interfaces) for all PSBs can be calculated by traversing Relationship (5) and (6). If afterwards the OutISB's filter list is empty (which must coincide with having no relations to RSBs), remove the OutISB and clear its relationship to Session.

### 3.2.3 Incoming Interface Merging

After a single or multiple (in case of RESV message processing) invocations of Outgoing Interface Merging, all PHopSBs that are marked for update are subject to Incoming Interface Merging. During this sequence, it is again possible to traverse relationships, instead of collecting state blocks. The details of this merging operation depend on the filter style for the session. In case of distinct reservations (FF), each PSB that relates to the PHopSB is considered separately. All OutISBs accessible through this PSB are merged and a flow descriptor is created, containing the PSB's sender address and the merged FlowSpec. For shared reservations, all OutISBs having a relationship to any of the PSBs are merged and the resulting flow descriptor contains the set of all sender addresses and the single merged FlowSpec.

### 3.2.4 Timeout Processing

According to the soft state paradigm, each state block is associated with a timer and deleted upon timeout. Periodic refresh messages restart the timer. Timers are directly connected to the object they apply to and the actions resulting from a timeout are similar to those when receiving a PTEAR or RTEAR message. The only difference is that the respective PTEAR/RTEAR message has to be created instead of just forwarding it.

## 4 Software Design

Given the objectives of the project, these goals have been set for an implementation:

- Message handling (creation/interpretation) should be clear, simple and extensible.
- Message processing should be clear and comprehensible, yet efficient.
- The implementation should be portable, but also nicely integrate with system level interfaces.

The design that has been chosen is a hybrid form of object orientation and procedural design. Object orientation does not seem to be fully appropriate for implementing state machines like network protocol engines, however, many aspects of an implementation can benefit from data encapsulation, inheritance and polymorphism. C++ has been selected as the programming language of choice to implement such a hybrid design under the given objectives. In the following, identifiers stemming from the implementation are printed in *italic*, when they are introduced. Figure 2 gives an overview of the design.

In this picture, the main components, which together form the contents of a global *RSVP* object, are shown. An RSVP object represents an RSVP-capable router and interacts through abstract interfaces with system-dependent services like routing, network I/O, traffic control and others. Multiple *Session* objects exist, representing currently active RSVP sessions. A number of *LogicalInterface* objects encapsulate physical and virtual interfaces of the underlying system. Logical interfaces are numbered and the number is used as LIH (Logical Interface Handle, see [2] for details). The API is modelled as a dedicated object, called *API\_Server*, containing a special instance of class *LogicalInterface*, and all information about currently active API clients. RSVP messages are encapsulated in a *Message* class and passed between *LogicalInterface* and *Session* objects, potentially involving *API\_Server*. Global state is kept in the RSVP object, for example, the current message, a PHopSB refresh list, etc.

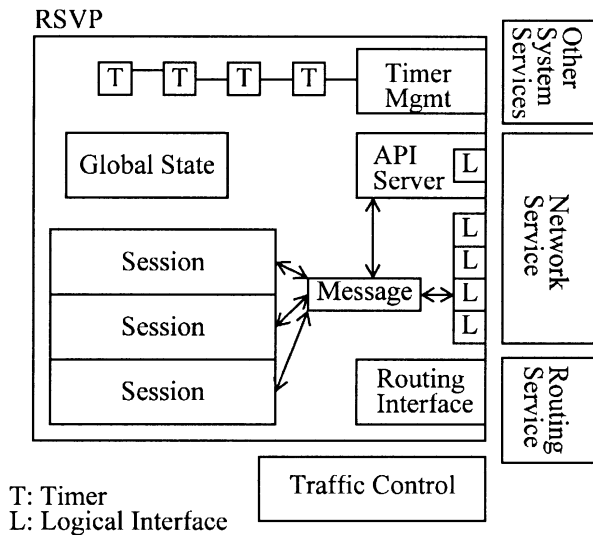


Fig. 2. Design of RSVP Implementation (Overview)

#### 4.1 Message Processing

Each incoming message arrives at the main RSVP object. After preprocessing and updating global state, the message is dispatched to the appropriate *Session* object for further processing. Some of the message processing rules from Section 3.2.1 are carried out in the RSVP object (e.g. finding or creating a *Session* object), but the majority is implemented in class *Session*.

The sequence *Outgoing Interface Merging* is link-layer dependent and consequently, functionality is split up. Common merging logic is implemented in class *OutISB*. A base class *TrafficControl* provides basic services and a uniform calling interface for the link-layer specific part. This calling interface takes an instance of *OutISB* and potentially a list of newly arrived filters as input parameters. All state that

is needed for admission control and updating of the underlying scheduling system is then accessible through OutISB. Both classes TrafficControl and OutISB are inherited by link-layer specific classes.

Incoming Interface Merging takes place when reservation state has changed, that is, if FlowSpec or FilterSpec of an OutISB has been modified. It is implemented in class RSVP, so that it can directly access all relevant global state information.

## 4.2 Implementation Details

**Relationship Representation.** Relationships are implemented as dedicated classes, which are used as base classes for those classes they apply to. The relationship classes automatically maintain referential integrity. A single-object relationship is internally represented by a pointer or reference, whereas a multi-object relationship is internally represented by a sorted list of pointers to the respective objects. Relationship (5) is internally stored as an array of pointers, because at most one OutISB exists at each interface and can be accessed directly by using the interface's unique LIH as index.

**Timers.** Timer management is logically separated from the rest of the implementation, such that it can be independently optimized without considering other parts of the code. A base class *BaseTimer* exists, from which refresh and timeout timers are derived. They are controlled by their owners, but handled commonly through BaseTimer. Currently, all timers are kept in a container, ordered by their expiration time. This design completely hides implementation details between timer management and timer clients.

**Container Classes.** A simple container library for lists and sorted lists has been implemented, in a style similar to the C++ STL (Standard Template Library). While it is conceptually very advantageous to use common container classes, it seems not necessary to provide the most efficient implementation for them. It is left to the user of this implementation to decide whether utmost efficiency is required when accessing certain containers or not. Because of the encapsulated design, testing of different algorithms and data layouts for containers is possible with relatively low effort.

## 4.3 Lessons Learned

It seems clear that introducing PHopSB and OutISB as important central state blocks representing merging state provides advantages due to their naturally given relations to RSBs and PSBs. The notion of recalculating relationships at every stage of message processing seems sub-optimal compared to maintaining and traversing these relationships. Some additional details are listed in [3].

## 5 Implementation Status

In this section, the current implementation status is described in comparison to the RSVP specification. This implementation is a full implementation of RSVP

operations, except certain limitations given below. It is developed and tested to automatically compile on Solaris 2.6, FreeBSD 3.X and Linux 2.X operating systems, using GNU C++ 2.95 and higher. The complete source package consists of approximately 19,000 lines of code. System-dependent code is cleanly separated and consists of about 2,000 lines of code with at most 150 lines dedicated to each system. The software is publicly available from <http://www.kom.e-technik.tu-darmstadt.de/rsvp/>.

## 5.1 Features

The implementation already provides some features that are new to an RSVP implementation and rather rare for experimental signalling protocols in general.

RSVP can be run in an emulation mode, in which multiple daemons execute on the same or differing machines and use a configurable virtual network between them, including shared link media and static multicast routing. Without such a feature, examinations of RSVP protocol behaviour in non-trivial network topologies are only possible by using a simulator or by using real systems. In the second case, it is necessary to start multiple processes on multiple machines needing super-user privileges and a suitable infrastructure. The emulation mode allows to experiment without the need for additional software nor hardware. A test-suite can be created by writing high-level configuration files, from which detailed configuration files are built with a special tool. A preconfigured test-suite consisting of 16 virtual nodes and including test scenarios already exists. Furthermore, the emulation mode can be combined with real operation, for example, to test interoperability with other implementations.

Communication between RSVP daemon and API clients uses soft state. This is deemed useful in cases when RSVP operates on a router on behalf of an API client at a different host. There is no need for complicated connection management and the API can be treated similar to an ordinary RSVP hop. It is configurable at compile time to have asynchronous API upcalls realized by signals or by using threads. Many other options devised for testing purposes are configurable at compile time, as well.

## 5.2 Limitations

Some of the properties of a full compliant RSVP implementation are currently missing. The main reason for them to be missing is their relative importance with respect to the project goals, compared to the effort necessary to develop and test these features.

- IPv6 is currently not supported. Due to the modular and portable design of the software, this should not create too much effort, yet it has to be tested then.
- UDP encapsulation as described in [2] is not supported. It is not planned to support this in the future, because it does not belong to the core of the specification and it is already discussed in the IETF to drop this requirement [14].

### 5.3 Traffic Control Interface

An interface to real packet scheduling is provided for the CBQ package on Solaris and for HFSC and CBQ scheduling using the ALTQ package on FreeBSD (see [3] for appropriate references). In the absence of real scheduling packages, the total amount of available bandwidth can be configured per interface. For each reservation, the necessary resource requirements are calculated in terms of service rate and buffer. The results are checked against the available capacity and logged.

## 6 Summary and Future Work

In this paper I have presented an implementation of RSVP that is based on different design and implementation paradigms than existing work. The description of RSVP operations becomes more comprehensible when using object-relationships as principle method of describing state blocks and message processing. Furthermore, a high-level description of processing rules can be translated into implementation details and vice versa with less semantic deprivation. A brief specification of RSVP processing rules is presented to demonstrate the capabilities of this approach. Certain optimizations have been carried out and additional tuning seems possible, if an implementation is based on maintaining object-relationships instead of recomputing them when needed. Design objectives for an experimental RSVP platform have been formulated and a design for an RSVP implementation is presented, following these objectives and being based on object-relationships. To a large extent, the design objectives have been met by the prototype. It is shown in this paper, how an experimental research platform can benefit from the application of modern software principles. The implementation described in this paper will be publicly available to the research community.

There is still a lot of research work to be carried out in the area of signalling resource requirements. A formal specification of RSVP in terms of relational algebra could be derived from the results of this work. This in turn could be used for formal verification of protocol implementations and modifications. Many potential protocol refinements remain open for examination. For this project, it is planned to further extend and tune the implementation as well as completing to port it to a simulation environment, which is already under way. Additionally, the research issues and existing proposals that are mentioned in Section 1 are going to be explored based on this implementation. Finally, results from this project might be useful when new proposals for new signalling protocols are being discussed, implemented and tested.

### Acknowledgments

Jens Schmitt implemented the initial integration of the CBQ and ALTQ packages. I also acknowledge the help of Jens Schmitt and especially Nicole Berière during preparation of this paper.

## References

- [1] L. Zhang, S. Deering, D. Estrin, S. Shenker, and D. Zappala. RSVP: A New Resource ReSerVation Protocol. *IEEE Network Magazine*, 7(5):8–18, September 1993.
- [2] R. Braden, L. Zhang, S. Berson, S. Herzog, and S. Jamin. RFC 2205 - Resource ReSerVation Protocol (RSVP) – Version 1 Functional Specification. Standards Track RFC, September 1997.
- [3] M. Karsten. Design and Implementation of RSVP based on Object-Relationships. Technical Report TR-KOM-2000-01, Darmstadt University of Technology, February 2000. Available at <ftp://ftp.kom.e-technik.tu-darmstadt.de/pub/TR/TR-KOM-2000-01.ps.gz>.
- [4] USC Information Sciences Institute. RSVP Software, 1999. <http://www.isi.edu/div7/rsvp/release.html>.
- [5] M. Karsten, J. Schmitt, L. Wolf, and R. Steinmetz. An Embedded Charging Approach for RSVP. In *Proceedings of the Sixth International Workshop on Quality of Service (IWQoS'98)*, Napa, CA, USA, pages 91–100. IEEE/IFIP, May 1998.
- [6] M. Karsten, J. Schmitt, L. Wolf, and R. Steinmetz. Provider-Oriented Linear Price Calculation for Integrated Services. In *Proceedings of the Seventh IEEE/IFIP International Workshop on Quality of Service (IWQoS'99)*, London, UK, pages 174–183. IEEE/IFIP, June 1999.
- [7] M. Karsten, N. Berier, L. Wolf, and R. Steinmetz. A Policy-Based Service Specification for Resource Reservation in Advance. In *Proceedings of the International Conference on Computer Communications (ICCC'99)*, Tokyo, Japan, September 1999.
- [8] J. Schmitt, L. Wolf, M. Karsten, and R. Steinmetz. VC Management for Heterogeneous QoS Multicast Transmissions. In *Proceedings of the 7th International Conference on Telecommunications Systems, Analysis and Modelling*, Nashville, Tennessee, March 1999.
- [9] J. Schmitt, M. Karsten, L. Wolf, and R. Steinmetz. Aggregation of Guaranteed Service Flows. In *Proceedings of the Seventh International Workshop on Quality of Service (IWQoS'99)*, London, UK, pages 147–155. IEEE/IFIP, June 1999.
- [10] R. Braden and L. Zhang. RFC 2209 - Resource ReSerVation Protocol (RSVP) – Version 1 Message Processing Rules. Informational RFC, September 1997.
- [11] B. Lindell, R. Braden, and L. Zhang. Resource ReSerVation Protocol (RSVP) – Version 1 Message Processing Rules. Internet Draft, February 1999. Work in Progress.
- [12] J. Schmitt. Extended Traffic Control Interface for RSVP. Technical Report TR-KOM-1998-04, Darmstadt University of Technology, July 1998. Available at <ftp://ftp.kom.e-technik.tu-darmstadt.de/pub/TR/TR-KOM-1998-04.ps.gz>.
- [13] E. S. Crawley, L. Berger, S. Berson, F. Baker, M. Borden, and J. J. Krawczyk. RFC 2382 - A Framework for Integrated Services and RSVP over ATM. Informational RFC, August 1998.
- [14] R. Braden. RSVP/IntServ MIB issues, June 23rd 1998. Contribution to rsvp mailing list. Available from <ftp://ftp.isi.edu/rsvp/rsvp-1998.mail>.



# Indirect RSVP for Virtual Cluster Cellular Mobile IP Networks

Yu Zeng, Jon W. Mark, and Xuemin Shen

University of Waterloo, Waterloo, Ontario, Canada N2L 3G1  
{yzeng, jwmark, xshen}@bbcr.uwaterloo.ca

**Abstract.** Mobile IP (MIP) could be deployed in a wireless cellular network so that the integrated Cellular Mobile IP (CMIP) network can be used to support datagram delivery to mobile users. In order to provide QoS for realtime traffic in IP network, Resource reSerVation Protocol (RSVP) is needed to reserve network resources along the route which the datagrams follow. However, current RSVP fails to operate through the MIP tunnel. Moreover, the high registration rate in a CMIP network greatly degrades RSVP performance. In this paper, an Indirect RSVP (IRSV) over Virtual Cluster Cellular Mobile IP (VCCMIP) is proposed in which an assistant RSVP connection is introduced to assist the end-to-end major RSVP operation over the MIP tunnel. IRSVP signaling costs and packet loss rates are evaluated. Simulation results show that the proposed scheme can greatly increase RSVP performance in terms of packet loss and RSVP connection active percentage.

## 1 Introduction

The importance of Internet to present day society hardly needs reiteration. As the backbone network to support user roaming, it does create many challenging problems. In a hybrid wireless/IP-based network, connection establishment and provision for continuous communications between mobile hosts across the internet require unique identification of Internet Protocol (IP) addresses before any actual communication can begin. However, in an environment where the mobile hosts constantly change their access points, an interworking infrastructure and a networking protocol are needed to support mobility without disrupting any ongoing communication. Unfortunately, the current suite of internet protocols (TCP/IP) fall short of mobility support since they are designed under the assumption that the end hosts are fixed. A Mobile IP (MIP) [1]-[2] protocol has been specified by the Internet Engineering Task Force (IETF) to support host mobility in an IP network. It enables a mobile host to have two IP addresses, one for identification and the other for routing. MIP can be implemented over current cellular networks, namely Cellular Mobile IP (CMIP), to extend IP services to mobile users. In CMIP, base stations (BSs) serve as MIP home or foreign agents to assist packet forwarding. However, high mobility tends to trigger frequent IP address change and MIP registration update.

IP network provides “best effort” services. It does not guarantee Quality of Service (QoS) of multimedia traffic. In order to meet the pre-designed QoS required by realtime traffic, Resource reSerVation Protocol (RSVP) [3]–[4] is proposed to reserve network bandwidth along the route which IP packets follow. However, RSVP fails to reserve end-to-end resources from the correspondent host to the mobile host in a CMIP network because of the presence of MIP tunnel. Either a new scheme or a modification to RSVP is needed to operate over the CMIP network.

In this paper, an Indirect RSVP (IRSV) scheme for a Virtual Cluster Cellular Mobile IP (VCCMIP) network is proposed. This scheme enables RSVP operation over a CMIP network, and achieves low registration rate. The remainder of the paper is organized as follows. Section 2 gives a brief overview of MIP, CMIP and RSVP. The IRSVP and VCCMIP concepts are presented in Section 3. Simulation results and discussions on the performance of the proposed scheme are given in Section 4, and concluding remarks are given in Section 5.

## 2 Overview of MIP, CMIP and RSVP

### 2.1 Mobile IP

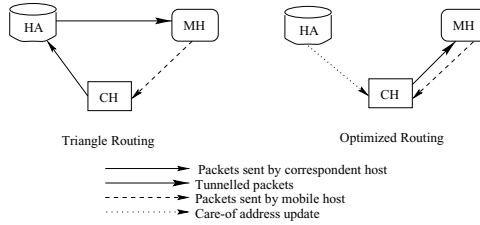
Several entities and addresses are defined in MIP. They are

- *Mobile Host (MH)* — A host that can migrate from one network to another without changing its IP address, while maintaining any ongoing communications.
- *Home Agent (HA)* — A router with an interface on the MH’s home network; the MH must update its home agent of its current location as it roams.
- *Foreign Agent (FA)* — A router with an interface on the MH’s visited network which assists the home agent to deliver datagrams to the MH while it is away from its home.
- *Correspondent Host (CH)* — A host that an MH is communicating.
- *Home Address* — An IP address assigned to an MH for permanent identification. It remains unchanged regardless of where the MH is located.
- *Care-of Address* — An IP address that is temporarily assigned to an MH when it is away from its home network. The MH may change its care-of address when it roams across the internet.

Mobile IP operates in the following way: home and foreign agents make themselves known by sending agent advertisement messages. Upon receiving an agent advertisement, the MH determines whether it is on its home network or a foreign network. An MH basically sends and receives packets like any other fixed host on its home network when it is at home. When the MH is away from its home network, it obtains a care-of address on the foreign network for the agent advertisement. The MH registers each new care-of address with its HA, possibly via its FA.

There are two types of routing in MIP as shown in Fig. 1.

- *Triangle Routing* — Packets sent by a CH to an MH connected to a foreign link are routed first to the MH’s HA and then tunneled to the MH’s current care-of address using IP encapsulation. On the other hand, packets sent by the mobile node are routed directly to the correspondent.
- *Optimized Routing* — The HA informs the CH of the MH’s care-of address and have the packets tunneled directly to the MH, bypassing the HA.



**Fig. 1.** Triangle routing vs. optimized routing

## 2.2 Cellular Mobile IP

CMIP implements MIP over a cellular network in the way that each BS acts as a HA or FA to assist packet forwarding. It is assumed in the CMIP network that:

- each and every MH, whether it is physically served by the wireline backbone IP network or the wireless cellular network, is identified by a unique IP address. The same rule applies to the CH.
- The CMIP network supports packet data communication between BSs. BSs are also identified by IP addresses.
- HA and FA have the same functionalities as defined in MIP. MIP registration occurs whenever an MH moves out of a BS service region.

## 2.3 RSVP

RSVP is a simplex, receiver-oriented resource reservation setup protocol designed for an Integrated Services Internet (IntServ). RSVP is used by a host to request specific QoS from the network for particular application data streams or flows. RSVP requests generally result in resources being reserved in each node along the data path. RSVP reserves resources for RSVP sessions. An RSVP session is defined as

$$\langle session \rangle ::= \langle DstAddr, PID, DstPort \rangle,$$

where *DstAddr* is the destination IP address of the data packets, *PID* is the IP protocol ID, and *DstPort* is the destination UDP/TCP port. RSVP signaling messages are sent hop-by-hop between RSVP-capable routers as raw IP datagrams with protocol ID 46, which is reserved for RSVP signaling. An RSVP message consists of a common header, followed by a variable number of “object”.

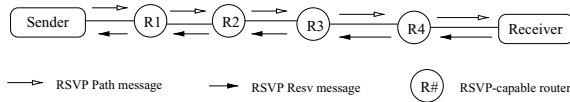
- *Common header*: The main purpose of the common header is to define the type of RSVP message. Seven types of RSVP messages have been defined so far, the most important two are Path and Resv messages, which will be discussed later.
- *Object*: Objects carry necessary information needed to perform RSVP signaling. For example, a  $\langle session \rangle$  object defines a particular session. *RSVP\_HOP* object carries the IP address of the RSVP-capable node that sends this message.

A typical RSVP message looks like

$$\langle RSVPMsg \rangle ::= \langle Header \rangle \langle Object1 \rangle \langle Object2 \rangle \dots$$

As shown in Fig. 2, a successful RSVP setup involves at least the exchange of two RSVP messages between a sender and a receiver, a Path message from the sender to the receiver to provide path information to all RSVP-capable routers, and a Resv message from the receiver to the sender to reserve network resources. Path and Resv messages are in the form of

$$\langle PathMsg/ResvMsg \rangle ::= \langle Header \rangle \langle Session \rangle \langle RSVP\_HOP \rangle \dots$$



**Fig. 2.** Resource reservation process

Suppose the sender in Fig. 2 wants to reserve resource to the receiver for its data flow, it sends a Path message

$$\langle PathMsg \rangle ::= \langle Header \rangle \langle RAddr, PID, DstPort \rangle \langle SAddr \rangle \dots,$$

where  $RAddr$  and  $SAddr$  are IP addresses of the receiver and the sender, respectively. When RSVP-capable router R1 receives this Path message, it

- caches path information carried by the Path message, and
- sends out another Path message

$$\langle PathMsg \rangle ::= \langle Header \rangle \langle RAddr, PID, DstPort \rangle \langle R1Addr \rangle \dots$$

to the next router.

The Path message is only used to carry RSVP path information to all routers along the data path. No resources are reserved at this stage. Once the receiver gets the Path message, it starts the reservation by replying with a Resv message

$$\langle ResvMsg \rangle ::= \langle Header \rangle \langle RAddr, PID, DstPort \rangle \langle RAddr \rangle \dots$$

to the sender. According to the pre-cached path information in those routers, this Resv message follows exactly the reverse path of the Path message. When R4 receives the Resv message, it reserves resources in itself accordingly and sends another Resv message to the next hop. After the Resv message is processed by all routers and the sender, resource reservation completes.

### 3 Indirect RSVP

Unlike the current IP network in which a host must have a fixed network interface in order to keep its ongoing IP connections alive, CMIP enables an MH to roam within the network while maintaining its IP layer connection so that any ongoing communication will not be interrupted. However, as illustrated in Section 3.1, RSVP over CMIP fails because of tunneling inherent in MIP. This deficiency can be rectified by isolating the tunnel and implementing RSVP over the tunnel separate from the rest of the return path. We refer to this mode of RSVP as Indirect RSVP (IRSV). The operation of IRSVP is described in Section 3.3.

#### 3.1 RSVP over CMIP

Since RSVP is a simplex protocol and CMIP has two routing schemes, several cases need to be studied separately as shown in Table 1.

**Table 1.** RSVP over CMIP operation cases

Packet direction	MH $\rightarrow$ CH	MH $\leftarrow$ CH
Triangle routing	<i>Case 1</i>	<i>Case 2</i>
Optimized routing	<i>Case 3</i>	<i>Case 4</i>

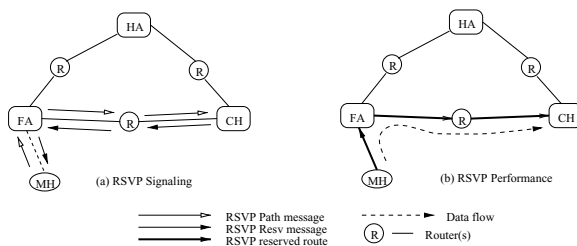
- *Case 1*: As shown in Fig. 3(a), to reserve resources from the MH to the CH, the MH sends RSVP Path message to the CH in the form of

$\langle PathMsg \rangle ::= \langle Header \rangle \langle CHAddr, PID, DstPort \rangle \langle MHHomeAddr \rangle,$

in which  $\langle CHAddr, PID, DstPort \rangle$  is the desired RSVP session. As mentioned before, this Path message is sent out as a raw IP packet with protocol ID 46. The intermediate RSVP-capable routers process the Path message accordingly. Upon receiving the Path message, the CH starts resource reservation by sending an RSVP Resv message

$\langle ResvMsg \rangle ::= \langle Header \rangle \langle CHAddr, PID, DstPort \rangle \langle CHAddr \rangle \dots$

Then a proper route as shown in Fig. 3(b) is reserved when the MH receives the Resv message. Data flows follow exactly the reserved route. Therefore, RSVP works well in *Case 1*.

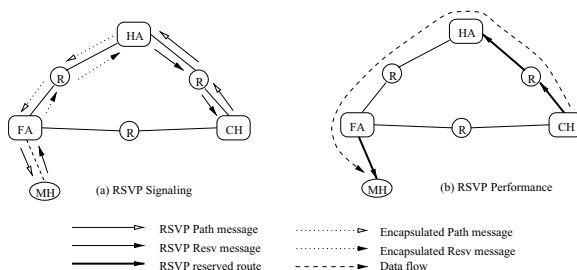


**Fig. 3.** RSVP over MIP: *Case 1*

- *Case 2*: Figure 4(a) shows RSVP signaling in *Case 2* in which the CH is the sender. The Path message sent by the CH,

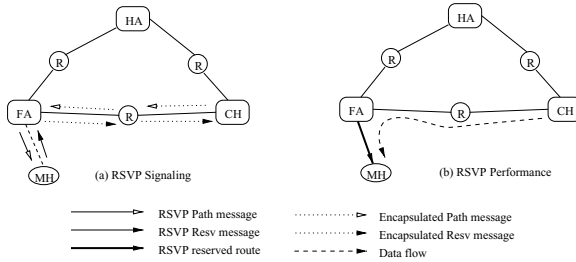
$\langle PathMsg \rangle ::= \langle Header \rangle \langle MHHomeAddr, PID, DstPort \rangle \langle CHAddr \rangle$ ,

can arrive at the HA properly. However, according to MIP, when the HA intercepts any packet destined to an MH, it encapsulates the packet in another IP packet with protocol ID 4 (reserved for IP encapsulation) and forwards it to the MH. The intermediate routers between the HA and FA identify RSVP Path message by checking whether the protocol ID is 46. As a result, these encapsulated Path messages are invisible to those routers so that no RSVP path informations are cached. In any case, this Path message can still arrive at the MH. The FA will be able to cache RSVP path information after decapsulation, but the previous RSVP\_HOP is the HA. The MH responds to the Path message with an RSVP Resv message. Upon the FA's receiving of the Resv message, an RSVP route is reserved from the FA to the MH. Since the FA will send Resv message to the HA according to the path information cached, no route is reserved from the HA to the FA, but the route from the CH to the HA is still reserved. The tunnel seems like a non-RSVP cloud. RSVP fails to operate over the MIP tunnel in this case as shown in Fig. 4(b).



**Fig. 4.** RSVP over CMIP: *Case 2*

- *Case 3* — *Case 3* operates the same way as *Case 1* since packets are routed the same way from MH to the CH no matter whether triangle or optimized routing scheme is used.
- *Case 4* — Similar RSVP signaling for *Case 2* happens to *Case 4*. RSVP fails to reserve resources from the CH to the FA as shown in Fig. 5.



**Fig. 5.** RSVP over CMIP: *Case 4*

In summary, RSVP works well in a CMIP environment to reserve network resources from the MH to the CH, but it fails in the opposite direction since the presence of the MIP tunnel makes RSVP Path message invisible to routers along the tunnel.

### 3.2 Mobility Impact

Mobility is another factor that degrades packet delivery performance in a CMIP network. When an MH moves to another BS/FA, it triggers the following events:

- *hard handoff*: The hard handoff involves the physical and data link layer connections to be established. Any upper layer communication cannot start until the connections are established.
- *registration and address update*: Once the MH detects that it has moved, it obtains a new care-of address, and starts an MIP registration by sending out registration request and waiting for registration reply. If optimized routing is used, care-of address in all possible CHs also need to be updated. Any IP layer communication cannot start until the above processes complete.
- *RSVP route update*: Although RSVP supports automatic adaptive route change, it takes time to update the new route. The worst case happens when there is address changes. Since RSVP session is identified by the triple: *DstAddr*, *PID* and *PortNum*, any address change will cause packets not to be recognized by RSVP-capable routers any more. Therefore, any address change results in the entire RSVP route being torn down and re-reserved, which will greatly degrade QoS of realtime traffic.

On the other hand, any new scheme can benefit from

1. reducing the handoff or registration rate, or both,
2. accelerating the registration and address update processes, and
3. introducing fast RSVP route update scheme.

The proposed IRSVP scheme improves packets delivery performance by taking advantage of item 1 and 3. The rest of this subsection discusses the proposed registration rate reduction scheme, while the next subsection describes the operation of the IRSVP scheme.

Although it is straight-forward in a CMIP network to construct FAs at BSs where wireless and wireline links conjunct, it endures high MIP registration rate because whenever handoff occurs, registration takes place, that registration rate equals handoff rate. In the proposed Virtual Cluster Cellular Mobile IP (VC-CMIP), MSCs act as FAs. All MHs associated with the same MSC are configured as one subnetwork, which includes several BSs. Handoff happens when the MH moves out of the service area covered by its current serving MSC. Therefore, registration happens at a much lower rate than handoff. The actual registration rate depends on the number of BSs controlled by an MSC, and the MH's mobility pattern.

### 3.3 Operation of Indirect RSVP

In IRSVP, the end-to-end RSVP connection is composed of two RSVP segments, a major RSVP connection excluding the MIP tunnel and an assistant RSVP connection through the tunnel. This section will focus on *Case 2* only because a similar approach can be implemented for *Case 4*.

As shown in Fig. 6(a), when an RSVP-capable HA receives a Path message from a CH, it

1. first encapsulates the Path message and forwards it to the MH to perform the major RSVP reservation,
2. then sends out a new RSVP Path message to the FA to start an assistant RSVP setup. This Path message is sent without encapsulation in the form of

$$\langle PathMsg \rangle ::= \langle Header \rangle \langle FAAddr, PID, DstPort \rangle \langle HAAddr \rangle \dots$$

*PID* and the *DstPort* are copied from the major RSVP Path message, while it changes the session by replacing the *MHAddr* with the *FAAddr*. The other fields of the Path message are set accordingly.

3. The HA also creates a session binding  $\langle MHHomeAddr, PID, DstPort \rangle \rightarrow \langle FAAddr, PID, DstPort \rangle$  so that the major RSVP session will be mapped to the assistant RSVP session.

When the FA receives the major and assistant Path messages, it

1. caches RSVP path information according to the major Path messages,
2. forwards the decapsulated major Path messages to the MH,



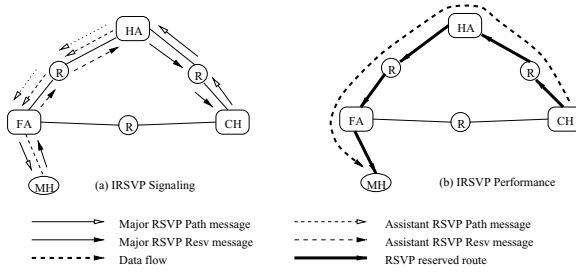


Fig. 6. Indirect RSVP

3. replies to the assistant Path messages with a Resv messages in the form of

$$\langle ResvMsg \rangle ::= \langle Header \rangle \langle FAAddr, PID, DstPort \rangle \langle FAAddr \rangle \dots$$

to reserve resources in the assistant RSVP segment, and

4. creates another session binding  $\langle FAAddr, PID, DstPort \rangle \rightarrow \langle MHHomeAddr, PID, DstPort \rangle$  so that the assistant RSVP session will be mapped back to the major RSVP session.

When the HA intercepts any RSVP session related data packets, it does not use IP encapsulation to route those packets. The HA simply changes the original destination IP address, which is the MH's home address, to the FA's IP address. This makes it possible for the intermediate routers along the tunnel to recognize RSVP sessions. In this way, the end-to-end RSVP connection is achieved by combining two separate RSVP connections with proper session mapping, as shown in Fig. 6(b).

## 4 Simulation and Discussion

### 4.1 Mobility Model

For simulation simplicity, assume a wireless service area is tilted by square cells. Each virtual cluster is composed of  $N$  radio cells. The MH mobility pattern is modeled as a Markov chain. Every  $T$  seconds, the test MH either stays with the same BS with probability  $P_{same}$ , or hands off to adjacent BSs with probability  $P_{other}$ , and

$$P_{same} + P_{other} = 1,$$

It is further assumed that if handoff happens, the MH is equally likely to hand off to any of the four adjacent cells. The MH registers its new care-of address with its HA only when it moves out of its current MSC service region. The network cost metrics used in the simulation for the performance evaluation are given in Table 2. Most of them are cited from [6] and [7].

**Table 2.** Network cost metrics

<b>Description</b>	<b>Value</b>
Cell size	800 <i>m</i>
Average speed of MHs	60 <i>km/h</i>
RSVP Path message process time	3 <i>ms</i>
RSVP Resv message process time	5 <i>ms</i>
Session mapping access time	3 <i>ms</i>
Tunneling and detunneling time	7 <i>ms</i>
Average Internet packet delay per hop	20 <i>ms</i>
Average wireless link delay	40 <i>ms</i>
To generate a control message	5 <i>ms</i>
Agent advertisement period	50 <i>ms</i>
RSVP Path and Resv message update period	30 <i>s</i>

## 4.2 Simulation Results

1. Registration rate — Figure 7 shows the registration rates in the CMIP and the proposed VCCMIP schemes when one MSC controls 4, 9 and 16 BSs. It is observed that there is significant registration rate drop in VCCMIP, which benefits realtime traffic.
2. Overall RSVP signaling cost — Figure 8 shows the overall RSVP signaling cost for RSVP-CMIP and the proposed IRSVP-VCCMIP schemes. The signaling cost is measured by the processing time needed to maintain an RSVP connection, which indicates the signaling complexity of the scheme. The time to generate control message, to process RSVP Path and Resv messages, and to access session mappings are considered in the simulation. Although it can be seen that the IRSVP scheme endures higher signaling cost because of the extra signaling cost associated with the assistant RSVP connection, the cost is still relatively small.
3. Percentage RSVP active time — Figure 9 shows the RSVP connection active percentage, which is defined as the percentage of time that the RSVP connection is alive, i.e., capable of transmitting packets. In the simulation, there are seven hops from the CH to the HA, from the HA to the FA, and two hops from the FA to the MH. It is worth to indicate that the IRSVP scheme greatly increases the RSVP active percentage, especially in a highly mobile environment (small  $P_{same}$ ).
4. Average packet loss rate — Figure 10 shows the packet loss rate vs. the RSVP Path message update period. Although short update periods brings low loss rates, it will sharply increase RSVP signaling cost. It is observed that IRSVP achieves much lower loss rate than RSVP. By comparing the two figures, it is also seen that low registration rate is important to obtain good performance.

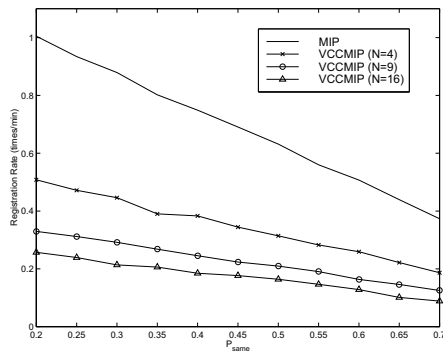


Fig. 7. Registration rate in CMIP and VCCMIP

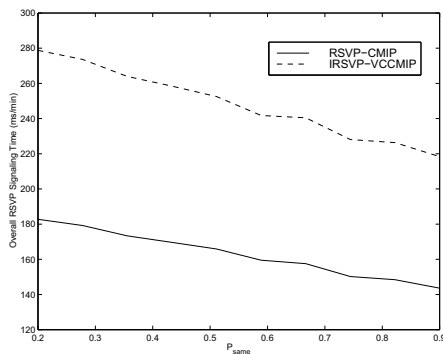


Fig. 8. Overall RSVP signaling cost

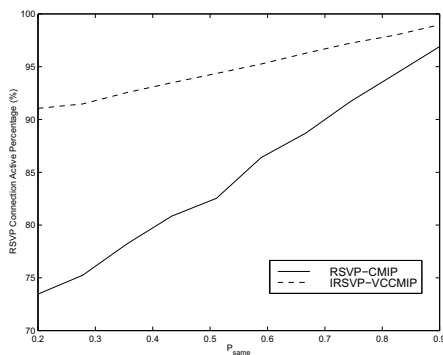


Fig. 9. RSVP connection active percentage

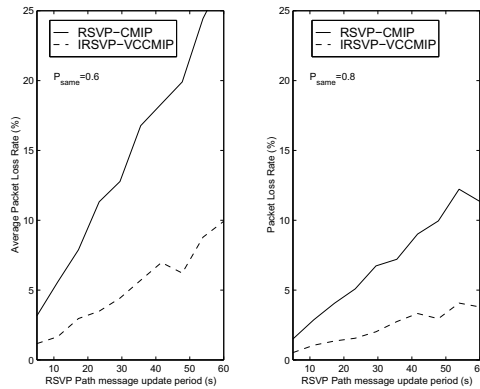


Fig. 10. Average packet loss rate

## 5 Conclusion

Since the RSVP-CMIP scheme fails to operate through the MIP tunnel and RSVP performance is greatly degraded in the presence of high registration rates, an IRSVP-VCCMIP scheme is proposed to address these problems. Analyses and simulation results show that the proposed scheme can increase RSVP performance in terms of packet loss and RSVP connection active percentage, while sustains an increase in signaling time. IRSVP performance may be further improved if a proper scheme is introduced in future work to accelerate the registration process, especially in the cellular network.

## Acknowledgments

This work has been supported by the Natural Sciences and Engineering Research Council (NSERC) of Canada under grant No. RGPIN7779.

## References

1. C. Perkins, "IP Mobility Support," *RFC 2002*, October 1996.
2. J. D. Solomon, *Mobile IP*. Prentice Hall, Inc., 1997.
3. R. Braden, L. Zhang, S. Berson, S. Herzog, and S. Jamin, "Resource ReSerVation Protocol (RSVP)–Version 1 Functional Specification," *RFC 2205*, September 1997.
4. P. P. White, "RSVP and Integrated Services in the Internet: A Tutorial," *IEEE Communications Magazine*, May 1997.
5. J. W. Mark, X. Shen, Y. Zeng, and J. Pan, "Issues in Wireless/IP Based Network," to appear in *IEEE 3G Wireless*, June 2000.
6. K. W. Ng and V. C. M. Leung, "An IPv6-based Location Management Scheme for Client-Server Computing over Mobile Data Networks," *IEEE WCNC*, 1999.
7. T. J. Kostas, M. S. Borella, I. Sidhu, G. M. Schuster, J. Grabiec, and J. Mahler, "Real-Time Voice over Packet-Switched Networks," *IEEE Network*, January/February 1998.

# Aggregation of Markovian Sources: Approximations with Error Control

Marco Conti, Silvia Ghezzi, and Enrico Gregori

CNUCE, Institute of National Research Council  
Via S. Maria 36 - 56126 Pisa - Italy  
{Marco.Conti, Enrico.Gregori}@cnuce.cnr.it

**Abstract.** One of the most challenging applications for high speed broadband networks is VBR video-conference and movies. Markovian models are a flexible tool to characterize a wide set of multimedia sources, including VBR video traffic. Specifically, these models are efficiently used for the description of a single video source. Nevertheless, teletraffic studies with superposed markovian sources have a computational complexity that grows exponentially with the increase of the number of the sources and this reduces the usefulness of these models in such an environment. The problem of characterizing the traffic generated by  $N$  superposed independent sources is addressed in the paper. Specifically, in the paper we define an aggregation method for the description of this kind of traffic. The state space of our aggregate model strictly depends on the required level of accuracy. Furthermore, we can decide a-priori the complexity of the aggregate model as function of the precision we want the model to have.

## 1 Introduction

One of the most interesting and emerging application is Variable Bit-Rate (VBR) video. This application is characterized by both a complex traffic profile and stringent QoS requirements (low packet-loss rate and low transmission delays). To efficiently provide the QoS required by these kind of applications, a very accurate video traffic model is needed. For this reason the modeling of VBR video sources has recently received significant attention. Since MPEG coding algorithms are becoming the standards for VBR video coding [2], in this paper we will model VBR video sources as MPEG sources. The model has been proposed, utilized and validated for describing VBR video sources, coded by an MPEG coder.

At the present, there is currently not a widely accepted model and three types of models have been proposed for VBR sources: autoregressive models ([12], [14]), Markov chain based models ([4], [13]) and models which capture properties of self-similarity and long-range dependence [7]. Although Markov chains do not have long-

---

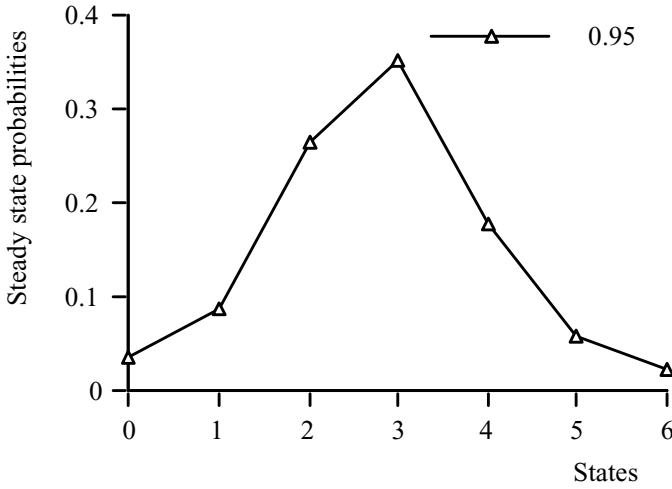
Work carried out under the financial support of CNR, in the framework of the project "Sistemi radiomobili multimediali nell'evoluzione verso UMTS".

range dependence as self-similar models, in [10] it is shown that VBR video sources are adequately modeled by a Markov chain as long-range dependence is not a crucial property in the teletraffic studies of interest. Furthermore, Markov models have been shown to provide an adequate traffic characterization for statistical multiplexing studies. Therefore in this paper we assume the use of Markov chains to predict the statistical properties of VBR video sources.

In this work, we start from a markovian characterization of a single source. Then we present and validate a method for the construction of a Markov chain describing the traffic generated by a set of superposed *i.i.d.* sources. We first show an *exact model* for the description of the traffic generated by  $\bar{H}_k = [H_1^{(k)}, H_2^{(k)}, \dots, H_N^{(k)}]$  superposed sources, which consists of the simple overlap of  $N$  single-source traffic models. In this paper we propose an approximate *aggregate model* to characterize the traffic generated by  $N$  superposed sources. Specifically, we develop, by using the single-source model, a multiple-source model based on the aggregation of the output traffic of  $N$  superposed video sources. The aggregate model accomplishes a balance between the complexity and the accuracy level the model will have. Later on, we give a comparison between this aggregate approach and the exact one by measuring the basic statistics such as mean bit rate, standard deviation, distribution and autocorrelation function.

## 2 Markovian Modeling of VBR Video Sources

The aim of this work is to find a suitable characterization of the video traffic generated by  $N$  independent and identical sources. Analog video signals are digitized and compressed before being transmitted over communication networks. Current compression techniques usually yield VBR video traffic. In this model the process  $\{H_k | k \geq 0\}$  describes the total amount of bits generated by a VBR source in a time unit. To avoid unnecessary complexity (in the state space of  $\{H_k | k \geq 0\}$ ), the bit rate information is quantized into a number  $M$  of levels, and the quantization levels are defined in a uniform way: the possible bit rate per time unit  $H_k$  is included in a range delimited by a minimum value  $l$  and a maximum value  $u$ ; the quantization levels are obtained by dividing the range  $u - l$  per  $M$ . The transition probabilities  $P\{H_k | H_{k-1}\}$  are computed through a movie trace and occur every time unit. The structure of the resulting Markov transition matrix is tridiagonal, or block tridiagonal [1], and this structure attests the clustering of bit rates within a scene, while the number of states in the model indicates the range of scene types. Furthermore, transition probabilities are concentrated on the diagonal (0.95) that indicates a long sojourn time in a single state and a high value of correlation, which are typical characteristics of markovian chains in the modeling of VBR video sources [4]. The steady state probabilities for the bit rate  $H$ , with a value of  $M$  equal to 7, are shown in Figure 1.



**Fig. 1:** Density of the steady state probability

This is a general model and one of its most practical utilization is the characterization of sources coded according to an MPEG encoder. An MPEG encoder produces a pattern that is periodically repeated. This pattern is referred to as GOP (group of picture) and it often made up of 12 frames. Previous studies [4] show that this single-source model is able to capture the properties of a VBR source such as steady-state probabilities and basic statistics. The auto-correlation of the GOP sequences generated by the model has a very slow decay. In the following, we use this single-source markovian model to design a model for the characterization of  $N$  superposed *i.i.d.* sources.

### 2.1 Exact Model

We recall that the system we aim to describe consists of  $N$  *i.i.d.* sources, that is we have  $N$  traffic flows, each of them generated according to the same distribution and in an independent way. The straightforward way to describe the  $N$  sources is to compute a transition probability matrix obtained from the Kronecker product of the  $N$  single-source Markov chains. The final matrix should contain the union of all possible states in which the  $N$  sources can pass.

Let  $N$  be the number of sources, each of them is described by its proper bit rate per time unit  $H_k$  ( $k = 1, \dots, N$ ). For a  $N$ -source system, if we choose to quantize the bit rate  $H_k$  in  $M$  levels then the state space of the resulting Markov chain consists of  $M^N$  possible states. In this paper to produce numerical results we assume  $M = 7$ .

The exact model describes the  $N$  video-sources system by representing the characteristics of each source in isolation, but the state space of the Markov chain increases exponentially with the number of sources, because of the Kronecker product.

The dimension of the transition probability matrix ( $7^N \times 7^N$ ) makes this approach useless when the number of sources grows. When we have i.i.d. sources the complexity of the aggregate source is lower than that obtained via the Kronecker product as state which are simple permutation can be merged together. In this paper we propose an alternative characterization method that further reduces the state complexity. In the following, we refer to the Kronecker-product model as *exact model* to distinguish it from the model we propose to solve the complexity problem. Our methodology is based on the definition of an aggregate model as expressed in the next section.

## 2.2 Aggregate Model

In the existing literature, several methods have been proposed in order to reduce the complexity of stochastic models. Among them, we focus on *aggregation techniques* where the original system model is replaced with a reduced model which mimics the behaviour of the original one [5]. According to this method we aim to characterize the system by having a single stochastic aggregation process able to exhibit the same behaviour as several i.i.d. stochastic processes. The aggregate process is *flow equivalent* to the  $N$  superposed i.i.d. processes [11], that is at any time unit the traffic arrival rate of the aggregate process is the same than the one of the  $N$  processes together, but it requires less information with respect to the exact one.

Starting from the real system, whose space state consists of all the possible  $N$ -tuples  $\bar{H}_k = [H_1^{(k)}, H_2^{(k)}, \dots, H_N^{(k)}]$ , we aim to design an aggregate model whose states are obtained by grouping together  $N$ -tuples that are equivalent from both the total bit rate and their future evolution standpoint. Specifically, if two  $N$ -tuples  $\bar{H}_i = [H_1^{(i)}, H_2^{(i)}, \dots, H_N^{(i)}]$  and  $\bar{H}_j = [H_1^{(j)}, H_2^{(j)}, \dots, H_N^{(j)}]$  represent two states of the real system where the  $N$  sources  $i$ ) generate the same total bit rate per time unit and  $ii$ ) follow the same probabilistic evolution, then we want to collect these  $N$ -tuples in the same state of the aggregate model. Thus, while the state space of the exact model consists of  $7^N \times 7^N$  states, the aggregate model will have a reduced number of states and the dimension of the aggregate model state space depends on the level of approximation and the computational cost we want to have.

To summarise, the basic idea of the aggregate model is to group original states together taking into account the total bit rate and the evolution of the  $N$ -tuples in system. However, while it is easy to understand how to estimate the total bit rate corresponding to an  $N$ -tuple, it is not straightforward to define the rule which assesses the evolution of an  $N$ -tuple in the system.

## 2.3 Maintaining the Autocorrelation of the System

To better explain why it is important to group together states with the same evolution in the system, let us see an example.

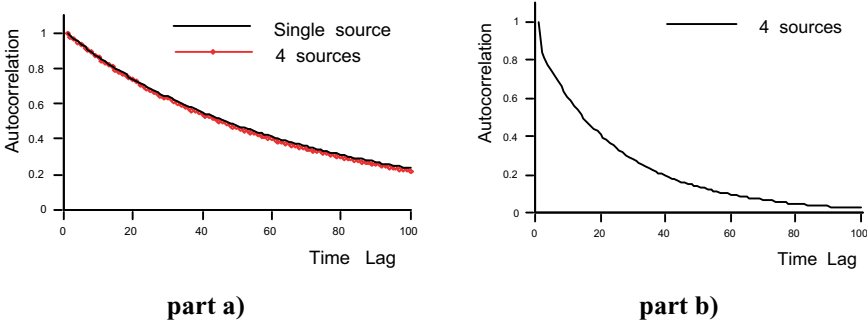


Let us assume to group together in a single aggregate state the states  $\bar{H}_k = [H_1^{(k)}, H_2^{(k)}, \dots, H_N^{(k)}]$  of the real system characterized by the same values of  $\bar{D}_q$ , where  $\bar{D}_q$  is the quantized value of  $\bar{D}_k$  and.

$$\bar{D}_k = \frac{\sum_{j=1}^N H_j^{(k)}}{N} \quad ;$$

for each state  $\bar{H}_k$  in the aggregate state characterized by  $\bar{D}_q$ .

It is evident that all the states  $\bar{H}_k = [H_1^{(k)}, H_2^{(k)}, \dots, H_N^{(k)}]$  with the same value of  $\bar{D}_q$ , are very similar from the traffic flow standpoint because they generate approximately the same total bit-rate at each time-unit (the approximation is due to the quantization of  $\bar{D}_k$ ). Anyway, some of the  $N$ -tuples  $\bar{H}_k = [H_1^{(k)}, H_2^{(k)}, \dots, H_N^{(k)}]$  belonging to an aggregate state can be different as regards their evolution in the system.



**Fig. 2.** Autocorrelation in the Exact Model (part a) vs Aggregate Model (part b)

We want to show now that if  $N$ -tuples  $\bar{H}_k = [H_1^{(k)}, H_2^{(k)}, \dots, H_N^{(k)}]$  are collected in an aggregate state only according to their total bit rate, then, when evolving, the aggregate state can generate a trajectory that is not compatible with the real system and that does not maintain its autocorrelation. For example, let a system be composed by 4 sources and let  $\bar{D}_i$  be quantized in 7 levels and suppose to adopt the uniform quantization strategy, as for  $H_k$ . Then, according to the aggregation and to the uniform quantization processes, the  $N$ -tuples [6600] and [3333] collapse into the aggregate state corresponding to  $\bar{D}_q = 3$ . These two  $N$ -tuples, corresponding to two states of the real system, are equivalent as far as the traffic flow, because they generate the same amount of traffic, but they are different as far as their trajectories. In fact, the sources in the  $N$ -tuple [3333] will probably pass in different states at the following transition step, by comparison with what will happen to the sources in the  $N$ -tuple

[6600]. This misconduct of the aggregate model can bring the model to a loss of autocorrelation because we merge states which should have different evolution to the same aggregate state, and this aggregation leads not to maintain the real correlation of the original system. Conversely, if we study the autocorrelation of the aggregate system when grouping together states like the two  $N$ -tuples [6600] and [3333], which are similar from the bit rate standpoint but different from the evolution standpoint, we have a very evident loss of correlation, see Figure 2 (part b). Thus, we can assert that this aggregate model is not able to maintain the autocorrelation in the system.

## 2.4 Distance Between States

In order to keep the autocorrelation as close as possible to the one of the real system, we want to group together only states that are equivalent from both the aspects we analyzed: traffic, i.e. rate per time unit, and autocorrelation.

The ideal situation is when an aggregate state only contains states which generate exactly the same total bit-rate per time unit and which evolve following the same trajectories at each transition step.

What we need first is a method able to estimate the *similarity* of two states in term of their evolution. After that, we only need to group together *similar* states. The basic idea is to identify a parameter able to capture this states' similarity. Since  $\overline{D}_q$  well describes the traffic behaviour we only add a new coefficient, called  $\varepsilon$ , which estimates the level of "precision" in term of ability of the model in capturing the similarity among states. An aggregate state in the new model is thus designated by the variable  $\{\overline{D}_q\}_\varepsilon$ . In order to give a better explanation of this concept we will give some definitions.

In the following, we will use " $N$ -tuple" at the same place as "state", and with "similar" we mean the similarity among states in term of similar trajectory in the system evolution.

**Definition 1.** A state  $\overline{H}_k = [H_1^{(k)}, H_2^{(k)}, \dots, H_N^{(k)}]$  is called a permutation of another state  $\overline{H}_j = [H_1^{(j)}, H_2^{(j)}, \dots, H_N^{(j)}]$  when the values  $H_1^{(k)}, H_2^{(k)}, \dots, H_N^{(k)}$  and  $H_1^{(j)}, H_2^{(j)}, \dots, H_N^{(j)}$  are exactly the same and only differ for the order in the two  $N$ -tuples respectively. For example, the  $N$ -tuple [1262] is a permutation of [6221].

**Definition 2.** An  $N$ -tuple  $\overline{H}_k = [H_1^{(k)}, H_2^{(k)}, \dots, H_N^{(k)}]$  is called to be in a lexicographic order if the values  $H_i^{(k)}$  are not increasing when the value of  $i$  increases.

For example, the lexicographic order of the  $N$ -tuple [1262] is [6221].

**Definition 3.** Given two  $N$ -tuples  $\overline{H}_k = [H_1^{(k)}, H_2^{(k)}, \dots, H_N^{(k)}]$  and  $\overline{H}_j = [H_1^{(j)}, H_2^{(j)}, \dots, H_N^{(j)}]$  in lexicographic order, then the *distance*  $d(\overline{H}_k, \overline{H}_j)$  between the two  $N$ -tuples is defined as:

$$d(\overline{H}_k, \overline{H}_j) = \sum_{i=1}^N |H_i^{(k)} - H_i^{(j)}|.$$

For example, the distance between the two  $N$ -tuple [6221] and [4440] is equal to 7.

**Lemma 1** *The distance between two  $N$ -tuples  $\bar{H}_k$  and  $\bar{H}_j$  which are permutation between each other is  $d(\bar{H}_k, \bar{H}_j) = 0$ .*

For example, the distance between the two  $N$ -tuple [6221] and [1262] is equal to 0.

The demonstration of this Lemma is straightforward if we first rearrange the two  $N$ -tuples in a lexicographic order.

**Lemma 2** *Two states  $\bar{H}_k$  and  $\bar{H}_j$  whose distance is  $d(\bar{H}_k, \bar{H}_j) = 0$  are equivalent from the evolution in the system standpoint.*

The demonstration of this lemma is straightforward because all of the  $N$  sources are *i.i.d.* .

It's worth noting that there is no difference both in term of generated traffic and in term of their probabilistic evolution between two  $N$ -tuples which are permutation one each other.

*Definition 4.* Two states  $\bar{H}_k$  and  $\bar{H}_j$  are defined *similar* when their distance is  $d(\bar{H}_k, \bar{H}_j) = 0$

It is now possible to define the key idea that underlines our aggregation methodology. The methodology aims to define each aggregate state  $I$  by collecting states  $\bar{H}_j$  with equivalent bit rate per time unit and with a distance  $d(\bar{H}_k, \bar{H}_j)$ , with any other  $\bar{H}_k$  in  $I$ , as low as possible. If this happens, we maintain the autocorrelation of the system because we group together states which are very similar from the evolution standpoint.

*Definition 5* Let  $I$  be an aggregate state, which groups together  $N$ -tuples  $\bar{H}_k$ . We define  $\varepsilon$  as an estimation of the maximum possible distance  $d(\bar{H}_k, \bar{H}_j)$  between each pair of states in  $I$ :

$$\varepsilon = \max \left\{ d(\bar{H}_k, \bar{H}_j) \right\} \forall \bar{H}_k, \bar{H}_j \in I \quad (1)$$

$\varepsilon$  varies in the range  $[0, N \cdot (M-1)]$ , where  $M$  is the number of quantization levels and  $N$  is the number of sources.

According to this definition we can design an aggregate model with a target level of precision, which is at least  $\varepsilon$ , and where a state is function of  $\varepsilon$  and  $\bar{D}_q$ . In Table 1 we show a tradeoff between the complexity of the state space of the aggregate model and the precision obtained by varying the value of  $\varepsilon$  and using the case study introduced in Section 2.3.2, i.e. 4 sources and 7 levels of quantization. The state space dimension for the exact model is 2401 states which reduces to 210 states by simply grouping permutations.

In the following we present the procedure that implements the aggregation mechanism, and then we propose an example of an aggregate model.

Procedure of the aggregation process, for a given value of  $\varepsilon$  :

1) For each possible state  $\bar{H}_k = [H_1^{(k)}, H_2^{(k)}, \dots, H_N^{(k)}]$  calculate the quantized value  $\bar{D}_q$ ;

2) By grouping together all states  $\bar{H}_k$  with the same value of  $\bar{D}_q$  create  $M$  subsets  $I$  of states;

For each subset  $I$  do:

3) rearrange each  $N$ -tuple  $\bar{H}_k$  in lexicographic order;

4) for each  $\bar{H}_k = [H_1^{(k)}, H_2^{(k)}, \dots, H_N^{(k)}]$  in lexicographic order, calculate the corresponding decimal representation  $dec = \sum_{i=1}^N H_i \cdot 10^{N-i}$ ;

5) Order each  $N$ -tuple  $\bar{H}_k$  in increasing order with respect to  $dec$ ;

6) Aggregate according to the value of  $\varepsilon$  :

6.1) for each  $N$ -tuple in  $I$  calculate  $d(\bar{H}_i, \bar{H}_{i+1})$  and  $d_s = \sum_i d(\bar{H}_i, \bar{H}_{i+1})$  while  $d_s \leq \varepsilon$ ; do

6.2) Let  $j$  be the index by which  $d_s = \sum_{i=x}^j d(\bar{H}_i, \bar{H}_{i+1}) > \varepsilon$ , then group together all states from  $\bar{H}_x$  to  $\bar{H}_j$ ,  $j = x$ , and  $d_s = 0$ ;

6.3) goto 6

Let  $M$  be the number of quantization levels of  $\bar{D}_k$ , the mean bit rate per time unit generated by a state  $\bar{H}_k = [H_1^{(k)}, H_2^{(k)}, \dots, H_N^{(k)}]$  (as defined in Section 2.3), and let  $N$  be the number of sources. Then the number of states of the real system is  $M^N$ . Furthermore, we choose to quantize the value of  $\bar{D}_k$  in a uniform quantization.

By looking at the aggregation algorithm, it is easy to note that there are two different aggregation levels, see Figure 3. The first one is performed at the step 2 of the procedure where we create  $M$  subsets of the original state space, according to the quantized value of  $\bar{D}_k$ . Hence, we obtain  $M$  aggregate states containing  $N$ -tuples characterized by the same value of  $\bar{D}_q$ . Then, at the step 6, we perform the second level of the aggregation process. This level is worked out by keeping into

of the aggregation process. This level is worked out by keeping into consideration the precision and the complexity degree we want to obtain. Note that according to the computation of  $d_s$  there are not any pair of  $N$ -tuples in  $I$  with a distance bigger than  $d_s$ . Hence, we first establish a value of  $\varepsilon$ , i.e. the upper bound on the *precision* the model will offer, and then we perform a further aggregation: for each of the  $M$  subsets, we create a further number  $x$  of subsets. In all of these  $x$  subsets all of the  $N$ -tuples have a maximum distance  $d_s$  from any other  $N$ -tuple in the same subset equal to  $\varepsilon$ . It is worth noting that  $x$  could be different for each of the  $M$  subsets and in each of the  $x$  subsets the real maximum distance between  $N$ -tuples could be less than the target maximum value  $\varepsilon$ . Let  $i$  be the index that allows us to select one of the  $x$  subsets, then an aggregate state is represented by the pair  $\{\overline{D}_q, i\}$ , where  $i = 0, \dots, x - 1$  and  $\overline{D}_q = 0, \dots, M - 1$ . In the following we refer to this model as  $\varepsilon$ -based aggregate model. Examples of use of this model can be found in [6].

**Table 1.** Complexity vs. precision in the aggregate model

Precision level $\varepsilon$	0	1	2	4	8	12	20
Number of aggregate states	210	136	106	66	43	32	20

Table 1 displays the state space dimension of the aggregate model for each level of precision  $\varepsilon$ . Of course the maximum degree of precision is reached when  $\varepsilon = 0$ , because we only group together permutations of states identical from the evolution and the traffic standpoint, but, in this case, we obviously get the highest number of states. Nevertheless, if we want to have a model with the lowest complexity (low number of states) we will aggregate states according to an high value of  $\varepsilon$ .

### 3 Autocorrelation Study in the Aggregate Model

To check if the aggregate model well captures the autocorrelation we plot the autocorrelation function for a system with 4 sources and 7 uniform quantization levels. In the Figure 4, we plot the autocorrelation function related to the aggregate model obtained by varying the value of  $\varepsilon$ . It easy to note that when  $\varepsilon = 0$  the autocorrelation strictly overlaps the one obtained with the exact model. Also when  $\varepsilon = 1$  the autocorrelation is still well captured. Then we show the autocorrelation obtained by reducing the precision level, i.e. with  $\varepsilon = 7$ , in order to see how the autocorrelation decreases when we introduce approximation in the model.

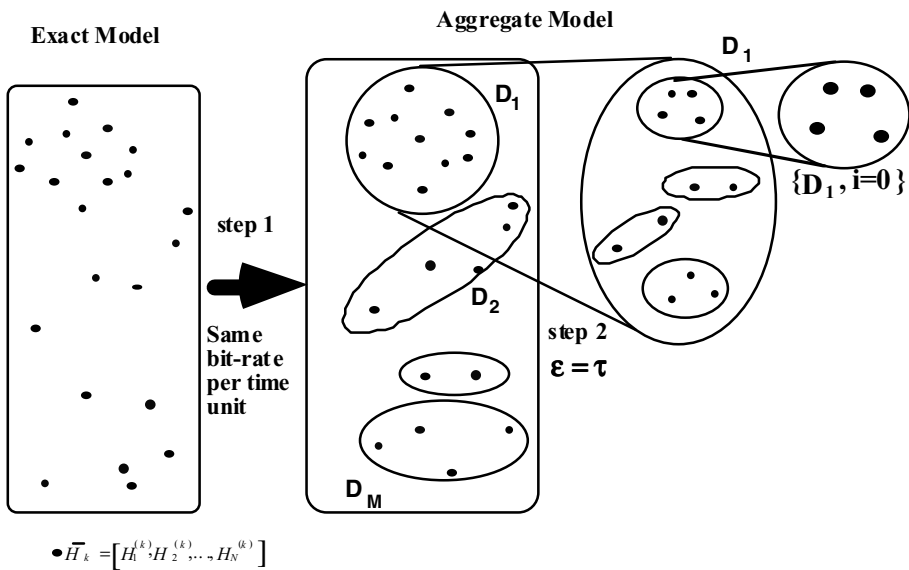


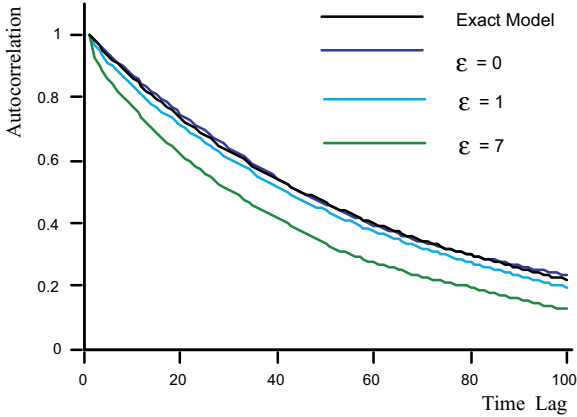
Fig. 3 Steps in the aggregation proces

We can see that, even in this case, the aggregate model is able to well capture the autocorrelation function: the loss of correlation is quite low comparing with the gain in term of space state complexity, see Table 2 for the number of states in the aggregate model. For all the other precision levels,  $1 < \varepsilon < 7$ , the autocorrelation is slowly getting worst as the value of  $\varepsilon$  increases, but always stays over the line plotted for  $\varepsilon = 7$ . Furthermore, to check the efficiency of the aggregate model in capturing the statistical behaviour of the real system, we show in the Table 2 the basic statistics. Results are obtained from a 1.000.000-steps long trace generated by the exact model and by the aggregate model. The statistics *Mean*, *Variance*, and coefficient of variation  $c$  correspond to the four sources.

Table 2. Comparison between basic statistics in the Exact model and in the Aggregate model

	Exact Model	Aggregate Model		
		$\varepsilon = 0$	$\varepsilon = 1$	$\varepsilon = 7$
number of states	2401	210	136	46
Mean	11.26367	11.31	11.313	11.29
Var	5.94	6.2	6.1335	6.2
c	0.21636	0.22	0.219	0.22

Table 2 shows that the aggregate model is able to capture the basic statistics of the system independently from the complexity and the precision level we decide to have. That is, any value of  $\varepsilon$  we choose, the model we get well approximates the real system.



**Fig. 4** Comparison between autocorrelation in the Exact model and in the Aggregate model

## 4 Conclusions

Modeling of superposed markovian sources is a difficult task due to state space complexity. In this paper, we have considered the modeling of  $N$  *i.i.d.* sources each characterized by the same Markov chain. We have showed a straightforward solution, *exact model*, consisting on the intersection of the  $N$  single Markov chains into one final Markov chain (obtained as the Kronecker product of the  $N$  single chains). The *exact model* is characterized by a state space which grows exponentially with the number of sources. To solve this problem we have proposed an alternative solution, *aggregate model*, whose key-point is the aggregation of the information of the original system, by providing a final Markov chain with a number of states lower than that of the exact model. Specifically, in the *aggregate model* we choose to describe the traffic flow generated by the system by using the information given by a pair of random variables  $\bar{D}_q$  and  $\varepsilon$  which aims to aggregate together states similar from the amount of arrivals generated per time unit and their evolution in the system standpoint.

The complexity of this approach does not depend on the number of sources but only depends on the level of precision required. A comparison between the *exact model* and the *aggregate model* have been carried out. The analysis of the basic statistics shows that the aggregate model is able to capture the characteristics of the traffic

generated by the superposed sources, by providing a very good fitting as regards the distribution figures, mean, standard deviation and the autocorrelation function. Furthermore a bound aggregation model has been proposed which gives an overestimation on the traffic generated by the sources, and which is suitable to design bandwidth allocation algorithm based on an upper bound rate rather than on the peak rate. Results demonstrate that in some particular video sources configuration the gain we get with a bound allocation is quite good.

## References

- [1] K. Chandra, A.R. Reibman, "Modeling One- and Two-Layer Variable Bit Rate Video", *IEEE/ACM Transactions on Networking*, Vol. 7, no. 3, June. 1999, pp. 398-413.
- [2] L. Chiariglione, "The development of an integrated audiovisual coding standard: MPEG", *IEEE Proceeding*, Vol. 83, No. 2, February 1995, pp. 151-157.
- [3] A. Chimienti, M. Conti, E. Gregori, M. Lucenteforte, R. Picco, "MPEG 2 Variable Bit Rate coding algorithm: analysis and modeling", *SPIE proceedings*, Vol. 2952, N. Otha editor, pp.597-607.
- [4] M. Conti, E. Gregori, A. Larsson, "A study of the impact of MPEG-1 correlations on video-sources statistical multiplexing", *IEEE Journal on Selected Areas in Communications*, Vol. 14, September 1996, pp.1455-1471.
- [5] M. Conti, S. Ghezzi, E. Gregori, "A Methodology for an Efficient Characterization of Superposed Markovian Sources", *Proceedings of IFIP ATM'98 Workshop on Performance Modeling and Evaluation of ATM Networks*, July 20th-22nd, 1998.
- [6] S. Ghezzi, "Methodologies for Efficient Characterization of Markovian Sources", Ph.D thesis (to be completed).
- [7] J.J Gordon, "Long range correlation in multiplexed Pareto traffic", in *Proceedings of the International IFIP-IEEE Conference on Broadband Communications*, Canada, 1996
- [8] D. Heyman, A.Tabatabai and T. Lakshaman, "Statistical Analysis and Simulation Study of Video Teleconference Traffic in ATM Networks", *IEEE Transactions on Circuits and Systems for Video Technology*, Vol. 2, No. 1, March. 1992, pp. 49-59.
- [9] D. Heyman, T. Lakshaman, "Source model for VBR broadcast-video traffic", *IEEE/ACM Transactions on Networking*, Vol. 4, Feb. 1996, pp. 40-48.
- [10] D. Heyman, T. Lakshaman, "What are the implication of long-range dependence for VBR video traffic engineering?", *IEEE/ACM Transactions on Networking*, Vol. 4, June. 1996, pp. 301-317.
- [11] K. Kant, "Introduction to Computer System Performance Evaluation", *McGraw-Hill, Inc.*
- [12] M. Nomura, T. Fujii and N. Ohta, "Basic Characteristics of Variable Rate Video Coding in ATM Environment", *IEEE Journal on Selected Areas in Communications*, Vol. 7, No. 5, June 1989, pp. 752-760.
- [13] P. Pancha and M. El Zarki, "MPEG Coding For Variable Bit Rate Video Transmission", *IEEE Communications Magazine*, May 1994, pp. 54-66.
- [14] G. Ramamurthy, B Sengupta, "Modeling and Analysis of a Variable Bit Rate Video Multiplexer", *Proceedings of IEEE INFOCOM*, Florence 1992, pp.817-827.
- [15] O.Rose, "Simple and efficient models for variable bit rate MPEG video traffic", *Performance Evaluation*, vol. 30, pp. 69-85, 1997.



# Multi-scaling Models of Sub-frame VBR Video Traffic

Iraj Saniee<sup>1</sup>, Arnold Neidhardt<sup>2</sup>, Onuttom Narayan<sup>3</sup>, and Ashok Erramilli<sup>4</sup>

<sup>1</sup> Bell Laboratories, Lucent Technologies, 600 Mountain Ave. Murray Hill, NJ 07974

<sup>2</sup> Telcordia, 331 Newman Springs Road, Red Bank, NJ 07701

<sup>3</sup> Physics Department, University of California, Santa Cruz, CA 95064

<sup>4</sup> Qnetworx, 1119 Campus Drive West, Morganville, NJ 07751

**Abstract.** In this paper, we investigate fine timescale (sub-frame level) features in video MPEG2 traffic, and consider their effect on performance. Based on trace-driven simulations, we demonstrate that short time features can have substantial performance and engineering implications. Motivated partly by recent applications of multi-scaling analysis to modeling wide area TCP traffic, we propose a multi-fractal cascade as a parsimonious representation of sub-frame traffic fluctuations, and show that it can closely match queueing performance on these timescales. We outline an analytical method for estimating performance of traffic that is multifractal on fine time-scales and long-range dependent on coarse timescales.

## 1 Introduction

VBR video traffic presents special challenges to the modeler. For one, it shows complex temporal structures with characteristic features on short (sub-frame), intermediate (1-100 frames) and long timescales ( $> 100$  frames). Secondly, traffic characteristics can depend sensitively on the coding scheme (e.g., H.261, MPEG) as well as specific parameters employed in the coding (as with MPEG2). Third, the high bitrates, relative to the link speeds currently deployed in networks, imply that aggregation levels in networks are too low to permit second-order descriptions (e.g., variances, autoregressions) that are complete (in the sense that two aggregate video streams with the same second-order descriptions can induce very different performance). While there has been considerable research in this area (for a representative, but far from exhaustive list see [1,2,3,4]) a definitive characterization that is robust (applicable across all coding parameters and schemes), comprehensive (covering all timescales of engineering interest), and practical (parsimonious and tractable) is still lacking.

In this paper we use MPEG2 video as representative of VBR video traffic, and consider one aspect in its modeling: traffic fluctuations at the slice, or the sub-frame level. Much of the prior literature (see [2,3]) is concerned with modeling fluctuations in video traffic at and above the frame level. In other words, the data sets analyzed consist of timeseries of byte or cell arrivals per frame and above. But as has been argued [1], performance can be very often determined

by fluctuations at the cell level, and as such, one would expect characterizations at the slice level to be more accurate than the frame level for many queueing scenarios. Typically, fine timescale features in traffic are less robust than coarse timescale fluctuations in that they can be readily modified through buffering or shaping, and are more sensitive to content, coding scheme and coding parameters. For this reason, our focus is on a structural methodology that can capture the full range of potential sub-frame characteristics. Our approach is partly motivated by recent developments in the modeling of data traffic.

It is now generally accepted that sufficiently aggregated network data traffic exhibits self-similar scaling over a wide range of timescales [5,3]. The performance and traffic engineering implications of this property have been extensively explored (see for example [6]), and a Gaussian self-similar traffic model, Fractional Brownian Motion (FBM), has been proposed as a parsimonious and tractable model of packet traffic [7]. However, there are many networking scenarios where the conditions for validity of the FBM model do not hold: for example, with wide area TCP/IP traffic, and VBR video traffic.

While it is inevitable that on the shortest time scales any packet traffic must be non-Gaussian and therefore non-FBM, the important feature in wide area TCP/IP and VBR video traffic is that, unlike earlier work on LAN traffic [6], this is found to have a significant impact on network performance [8]. Further, directly measured WAN traffic [9,10] as well as detailed simulations of TCP [11] have been claimed to show that there is a short time regime below the self-similar range over which there still exist compact representations for the traffic process: the self-similar or fractal characterization at longer timescales is generalized to a multifractal one. Such a representation is also found to be useful in understanding the queueing delays of WAN traffic, although alternative models can also be used [8].

It has been argued [9] that this complicated short time behavior is due to the manner in which TCP controls the load put by each individual source on the network. The fact that the transition between the short and long time behavior occurs at a timescale of the order of the round-trip time of a TCP segment lends support to this scenario. In this paper, we consider similar questions for the short-time structure of MPEG2 video data. Unlike the case for TCP/IP traffic, any such structure is intrinsic to the complex manner in which the data is encoded into frames and subdivided into slices (see [1]), rather than being generated by flow control mechanisms responding to network congestion. Thus the earlier work on TCP/IP traffic represents the case of a closed network, where feedback affects the offered load, while the video traffic studied in this paper is the complementary case of an open network. In this sense, the application of open loop, flow level models (such as FBM with multi-fractal extensions) may be even more appropriate for VBR video traffic than wide area TCP/IP traffic.

The work in this paper firstly shows that for MPEG2 traffic there is a clear separation between a short time and long time regime occurring approximately at the time scale corresponding to a single frame. Secondly, the short time scale features are shown to affect performance substantially: this is demonstrated by

aggregating the video trace to the time scale separating the short time and long time regime, and interpolating down to finer timescales in various ways, including a multifractal cascade. The multifractal description is found to be adequate to describe the short time behavior.

We also outline an analytical estimation procedure for the viable operating point for a network, if its traffic can be represented by a purely multi-fractal cascade on short timescales. Not surprisingly, we find that multi-fractal scaling exponents alone are not sufficient to describe the traffic: one needs to specify the *magnitude* of the fluctuations in (different moments of) the traffic. We fix these by assuming that the behavior at the transition from short to long timescales should ‘match on’ from both sides [12].

There is a large literature on the modeling of MPEG2 traffic given the importance of video in multimedia communications. These models take full recognition of the *details* of how various MPEG2 frames are generated and relate to each other. For example autoregressive models [4] use the correlation between the I, B and P frames and use a gamma function for a near fit to the variable components. We aim to show a different modeling approach that, while not explicitly modeling the workings of the frame generation, recognizes the qualitative impact of these mechanisms via the scaling within the resulting traffic. This idea is further elaborated in Sect. 3 where we show the match between the performance induced by traces of a queue fed by MPEG2 traffic, and a trace whose sub-frame characteristics are re-constructed by a multi-fractal model.

The MPEG2 trace we study consists of 6 minutes of gymnastics at the 1996 Atlanta Olympics containing 280,000 time slices of duration  $4/3$  ms each, with a minimum, mean and maximum of 15, 815 and 5514 bytes per time slice respectively [13]. Traces of varying durations from other events at the Olympics were also investigated, and show the same essential features. While our primary focus in this paper are the traffic sensitive features within a single stream, it is understood that network traffic consists of (perhaps) limited aggregates of such individual streams.

The rest of this paper is organized as follows: Sect. 2 provides a short background on mono-fractal and multi-fractal scaling, and characterizes the traffic, showing a transition point between short and long time scale behavior; Sect. 3 describes experiments, in which the measured trace is aggregated to this transition point, and then interpolated down to fine timescales in a variety of ways, with the simulated performance compared to that of the original; Sect. 4 considers queueing analysis to support capacity planning and admission control with multi-fractal based descriptions of the traffic; Sect. 5 summarizes our conclusions with suggestions for further work.

## 2 Characterization of Traffic

Since this paper proposes a multi-fractal cascade as a promising approach to modeling fine timescale features in VBR video traffic, we first summarize some of the basic concepts associated with multi-fractals. Consider a traffic arrival

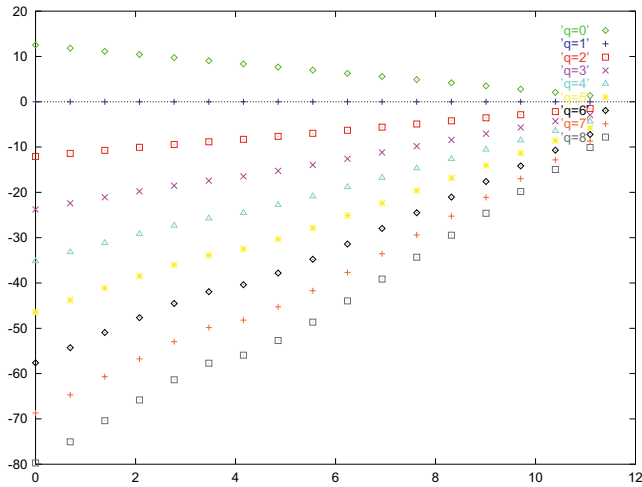
process  $A(0, t)$ , and its associated increment process  $X_\Delta(i)$ , defined by

$$X_\Delta(i) = A((i-1)\Delta, i\Delta) = A(0, i\Delta) - A(0, (i-1)\Delta). \quad (1)$$

The basic scaling hypothesis is that the moments of the increment process behave as:

$$\sum_i X_\Delta(i)^q \sim C(q) \Delta^{-\tau(q)} \quad \text{for } \Delta \rightarrow 0 \quad (2)$$

where the premultiplier  $C(q)$  is a scale factor that does not depend on  $\Delta$ . In practice, the scaling hypothesis can be said to be reasonable if the above behavior is satisfied over a *range* of timescales (for the processes considered in this paper, these would apply to the fine timescales). In general the *structure* function  $\tau(q)$  as defined above, if it exists, will be decreasing and nonlinear in  $q$ . When  $\tau(q)$  is linear in  $q$ , the scaling behavior is said to be *mono-fractal*.

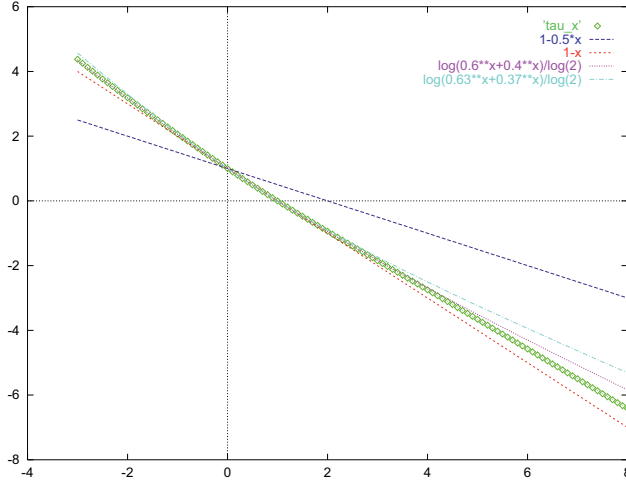


**Fig. 1.**  $\ln \sum_i X_\Delta^q(i)$  plotted for various values of  $q$  as a function of  $\ln \Delta$ . Here  $X_\Delta(i)$  is the amount of traffic arriving in the time interval  $[(i-1)\Delta, i\Delta)$ , with  $\Delta$  measured in units of the time interval between slices (1/750 seconds).

One of the standard techniques to generate multi-fractal scaling on fine timescales is the *cascade* construction. In the deterministic version of this approach, used later in this paper, a coarse timescale count over an interval is distributed over finer timescales by assigning a fraction  $p$  to the left half of the interval and a fraction  $1-p$  to the right half, with the parameter  $p$  characterizing the cascade. The process is repeated a number of stages, and the resulting process is marked by extreme irregularity over a range of fine timescales. There are numerous variations on this cascade construction [9]. Note that at an abstract level, the cascade construction does mimic the encoding action below a video

frame: the cells that constitute a frame are encoded into blocks, macroblocks and slices [1] which has the effect of distributing these cells over a frame duration in a bursty, irregular fashion.

Figure 1 shows the scaling of  $\sum_i X_{\Delta}^q(i)$  with  $\Delta$ . There is a transition from a short time regime at approximately 40ms (or 30 slices which correspond to 3.5 in the logarithmic time axis in Fig. 1). Interestingly, this is the time duration of a frame, suggesting that the short time behavior is characteristic of how MPEG2 distributes data between slices within a single frame.<sup>1</sup>



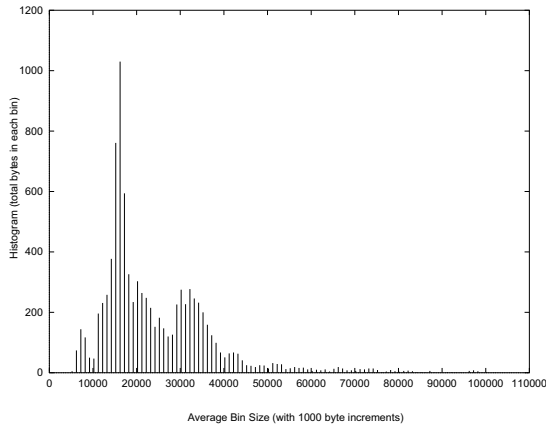
**Fig. 2.** The structure function,  $\tau(q)$  plotted against  $q$  for the original video trace (MPEG2) and for deterministic multifractal cascade interpolations of an aggregated version of the traffic, with  $p = 0.64$  (MFC 0.64) and  $p = 0.6$  (MFC 0.6). The aggregation interval is 40ms. Linear  $\tau(q)$ 's for FBM with  $H = 1.0$  (FBM 1.0) and  $H = 0.5$  (FBM 0.5) are also shown.

In the short time scale regime, the curves in the log-log plot of Fig. 1 are all linear, although admittedly over a not very large range of  $\Delta$ . The slopes of the straight lines fitted to the short time regime of the plots in Fig. 1 yield  $\tau(q)$  for various values of  $q$ . Figure 2 shows a plot of  $\tau(q)$  as a function of  $q$  obtained in this manner from Fig. 1, and is seen to be non-linear. By comparison, the familiar FBM arrival process can be said to exhibit mono-fractal scaling, when the fluctuations about the mean arrival rate (instead of the arrival rate itself) are considered:  $\tau(q)$  (for non-negative even  $q$ ) is equal to  $1 - qH$ , where  $H$  is the Hurst parameter characterizing the FBM. Thus the evidence for multifractal behavior at short timescales is i) the linearity of the plots in Fig. 1 at short

<sup>1</sup> The same threshold value of 40msec was observed in several MPEG2 traces that are characterized by a frame rate of 25 f/s, showing the connection between the observed fine time scale regime and sub-frame characteristics.

timescales ii) the nonlinearity of the resultant  $\tau(q)$  as a function of  $q$ , plotted in Fig. 2.

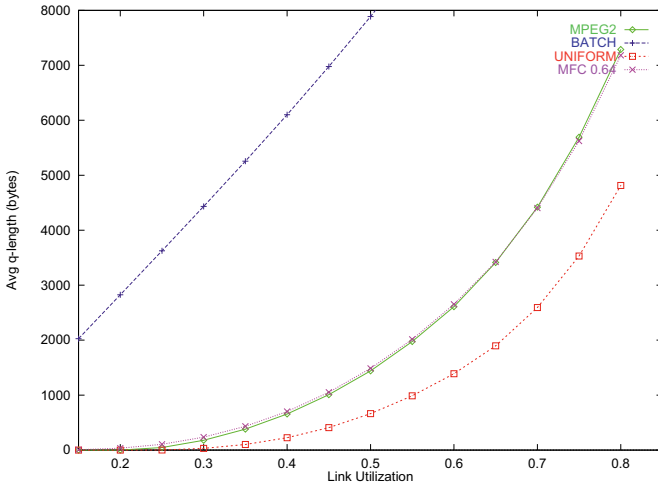
Since the traffic in different slices within a single frame is by no means random and uncorrelated, we do not expect traffic aggregated to the transition point between short and long time scale behavior (the frame level) to be Gaussian. This is borne out in Fig. 3. Aggregating a sufficiently large number of sources would result in Gaussian marginals.



**Fig. 3.** The marginals of the traffic aggregated to 40ms, showing that clear non-Gaussian behavior exists even at this level. The unnormalized probabilities are plotted, with bin sizes of 1000 bytes. Higher levels of aggregation of course eventually yield Gaussian marginals.

### 3 Performance Analysis

We now turn to the queueing behavior of the MPEG2 video traffic. Figure 4 shows a plot of average queue length in bytes vs. utilization for measured video traffic, obtained by running the trace through a simulated queue with capacity adjusted so that utilizations are varied from 10%-80%. Given that simulations are being done for a single video stream, a corresponding ATM networking scenario is: the VBR video stream is assigned to a single VC, and the capacity of the link corresponds to the bandwidth allocated to this VC in a per-VC queueing discipline. In most ATM switches, unallocated or unused capacity can also be used to serve this VC, so that the queueing backlogs studied here are upper bounds of the actual backlogs observed in a per-VC queueing system. The single stream simulation is also relevant for determining effective bandwidths, and for setting policing and shaping parameters. In this paper, the value of the single stream simulation is in identifying the statistical features of the traffic that determine performance. The average queue lengths increase sharply above 70%, and are significant even at the 50-60% utilization level. Interpreted as a per-VC



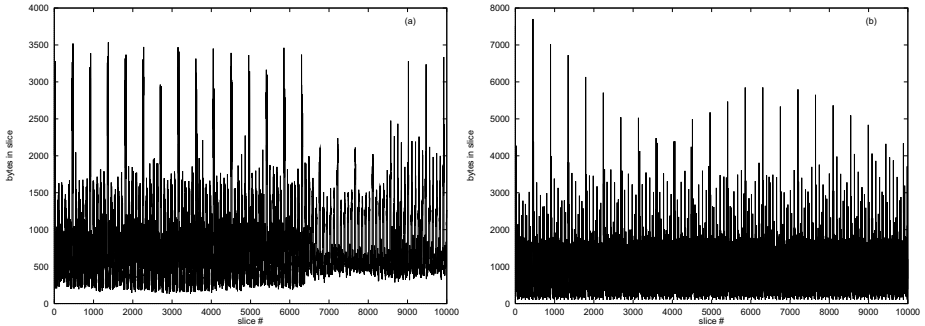
**Fig. 4.** Plot of the average queue length in bytes as a function of utilization for the measured video traffic (MPEG2), and various interpolations to an aggregated version thereof: i) a uniform interpolation (UNIFORM) of the aggregate traffic on time scales shorter than 40ms ii) all the traffic in any 40ms interval concentrated as a batch arrival (BATCH) and iii) a deterministic multifractal cascade (MFC 0.64) interpolation below 40ms, with  $p = 0.64$ .

backlog, the corresponding maximum queue lengths (not shown) of  $\sim 60\text{kB}$  lead to delays of 50 – 100ms, which is significant for MPEG2.

In order to find out how much the distribution of traffic at the sub-frame level affects the queueing delays, we aggregate the traffic to the frame level, and then distribute all the bytes in a single frame uniformly between the slices. Figure 4 shows the mean queue length as a function of utilization for such a smoothed version of the data, demonstrating that the delays are significantly underestimated in this approximation. As an alternative worst-case approach, we concentrate all the traffic in a frame at the beginning of the frame. This approximation is complementary to the previous one: the sub-frame level fluctuations are now maximized, whereas they were set to zero previously. Figure 4 also shows the mean queue length as a function of utilization in this worst case approach. The delays are now considerably overestimated.

We see that neither of these two interpolation schemes (used extensively in performance studies based on frame level measurements) works very well. While it is obvious that they should serve as lower and upper bounds to the actual performance, the wide gap between them shows the need for a realistic sub-frame description. In view of the results obtained in the previous section, it is not unreasonable to try a multi-fractal cascade interpolation. We consider the simplest deterministic multifractal cascade, where the traffic in any interval (starting from the highest level of aggregation, the frame level) is divided between the two halves of the interval with a fraction  $p$  always being apportioned to the

left half (and  $1 - p$  to the right half). Here  $p$  is a free parameter, which was adjusted to best match the delay versus utilization curve. The optimal value of  $p$  thus obtained was  $p = 0.64$ . The resultant mean queue length versus utilization is also shown in Fig. 4, showing remarkable agreement with the result for the original unaggregated data.

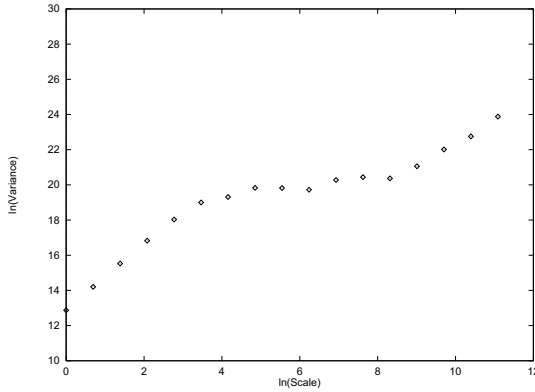


**Fig. 5.** Bytes per slice for (a) the original video traffic trace and (b) the original traffic aggregated to 40ms and then interpolated down using a deterministic multifractal cascade with  $p = 0.64$ . The level of irregularity in both plots is seen to be qualitatively the same.

Note that a multi-fractal interpolation with  $p = 1$  yields the batch construction and  $p = 0.5$  the uniform interpolation in Fig. 4. As an independent test of the choice  $p = 0.64$ , Fig. 2 shows the plot of  $\tau(q)$  as a function of  $q$  obtained for this interpolation. The curve compares reasonably to the  $\tau(q)$  for the original measured data, although the best agreement is actually obtained with a slightly lower value of  $p = 0.6$ . Also, Fig. 5 shows that the typical variability of the original trace is reproduced in the interpolated version.

It is pertinent to note that the MPEG coding scheme imposes cross-frame correlations in the traffic as well: I-frames, containing all the information in a frame occur at regular intervals, interspersed with the much more numerous P and B frames, which are much more compactly coded and therefore shorter. This is easy to see directly from Fig. 5, where the periodic structure of the high peaks (I-frame peaks) is apparent. Alternatively, this can be seen in the variance-time plot in Fig. 6, which clearly shows the three timescales mentioned at the beginning of this paper: i) the short time regime, which has been the focus of this paper ii) a very long time regime, where one obtains the long range dependence that seems ubiquitous for packet traffic (with a Hurst parameter  $H$  slightly greater than 0.6) and iii) an intermediate regime from roughly the frame level to the group of frames (GOF) level. The flat variance in this regime is because of anticorrelations between different frames in a GOF: a high information content I-frame is never followed by another I-frame. Note that there is no such intermediate time regime seen with TCP/IP traffic.





**Fig. 6.** Log-log plot of the variance of the traffic arriving in an interval of size  $\Delta$ ,  $\ln[E(X_\Delta^2) - E(X_\Delta)^2]$ , as a function of  $\ln[\Delta]$ , showing three different regimes.  $\Delta$  is measured in units of the time interval between slices (1/750 seconds).

Since the interpolation schemes discussed in this paper work with frame-level aggregates, they preserve all the correlations or anticorrelations present at higher time scales without trying to model them. A more ambitious approach might be to work with traffic aggregated to the group of frames (GOF) level, interpolate to the frame level keeping in mind the manner in which MPEG2 distributes the different frames, and then proceed to a sub-frame level using multifractal (or other) approaches. We leave this for future work; note that for queueing purposes, the different frames are treated identically.

## 4 Performance Analysis and Engineering

In the previous section we observed the significant performance impacts of the fine timescale fluctuations in the video trace, and the fact these could be modeled reasonably by a multi-fractal cascade. In the remainder of this paper we consider the implications of assuming that multiplicative cascades are indeed an appropriate way to characterize short- time features, and show how one can analyze the consequences for admissions control and capacity planning, using large-deviations theory.

Using standard large-deviations arguments [14], the loss probability at a link of capacity  $C$  and buffer size  $B$  that is driven by an arrival process  $A(t)$ <sup>2</sup> can be estimated by

$$P(Q > B) \sim \sup_{t>0} P(A(t) > Ct + B) \approx \sup_{t>0} \exp[-\Lambda^*(Ct + B, t)] \quad (3)$$

where  $\Lambda^*$  is the Legendre transform of the function  $\Lambda$ , defined as  $\Lambda^*(x, t) = \sup_s (sx - \Lambda(s, t))$ , and  $\Lambda$  is the logarithm of the moment-generating function

<sup>2</sup>  $A(t)$  is the function  $A(-t, 0)$  of (1) with the argument of the function simplified.

of  $A(t)$ . If  $A(t)$  is comprised of  $n$  independent streams  $A_I(t)$ , the function  $A$  can be expressed as [15]

$$A(s, t) = nA_I(s, t) = n \log \left[ E \left( \exp(sA_I(t)) \right) \right] = n \log \left[ \sum_{q=0}^{\infty} s^q E(A_I^q(t)) / q! \right]. \quad (4)$$

With a target loss probability of  $\varepsilon = \exp[-\lambda]$ , using (3) as an estimate of the loss probability yields the condition  $\lambda \leq \inf_{t>0} \sup_s [s(Ct + B) - nA_I(s, t)]$ . This can be inverted, to yield the maximum number  $N$  of sources that can be supported on the communication link:

$$N \sim \inf_{t>0} \sup_s \frac{s(Ct + B) - \lambda}{A_I(s, t)}. \quad (5)$$

Note that while estimates of performance measures such as loss rates can be far from their actual values using large-deviations-based methods, the corresponding engineering recommendations are typically more accurate. This is because near the operating point, small changes in traffic levels can have a large effect on the relevant performance measure.

Equation (4), together with the other equations in the previous paragraph, directly establishes the relation between multi-scaling and performance. One can use multi-scaling to parametrize the time dependence of the various moments in the sum in terms of a scaling exponent, and a constant pre-factor or intercept, rather than explicitly specifying the marginal distribution over a continuum of timescales. Equation (4) also demonstrates that a multi-fractal characterization in terms of  $\tau(q)$  is only “half-complete” in that it provides the scaling exponents, but not the constant pre-factors needed to evaluate it. In terms of the FBM model  $m, a, H$ , this is equivalent to providing the Hurst parameter, but not the peakedness parameter  $a$ . Accordingly, the procedure adopted here is to use as an approximation an FBM model for the coarser scales and then:

- finding analytical expressions for the scaling exponents of a multi-fractal cascade, which is feasible, at least for simple cascades;
- using the “boundary condition” of an FBM description over coarser timescales to determine the constant pre-factors.

Assuming that the cascade construction is applied at the level of individual streams, the moments of the MFC interpolated process over fine timescales are approximately given by:

$$E(A_{MFC}^q(t)) = E(A_{FBM}^q(\theta))(t/\theta)^{1-\tau(q)} \quad (6)$$

where we assume that an FBM description is valid for timescales greater than  $\theta$ , and a multi-fractal cascade is used below it. Equation (6) is the condition of continuity of  $E(A_I^q(t))$  across  $\theta$ . In terms of the parameter  $p$  of the semi-random MFC, the function  $\tau(q)$  is given by

$$\tau(q) = 1 + \log_2 \left[ \frac{p^q + (1-p)^q}{2} \right]. \quad (7)$$

The boundary condition at  $\theta$  is obtained from

$$E(A_{FBM}^q(\theta)) = \sum_{k=0}^{k \leq q/2} \binom{q}{2k} (m\theta)^{q-2k} (am\theta^{2H})^k (2k-1)!! \quad (8)$$

where we have used the fact that  $A_{FBM}(\theta) = m\theta + X_{FBM}(\theta)$ , with  $X_{FBM}$  Gaussian with zero mean. Thus, in principle, given a description of the traffic in terms of a coarse timescale FBM model, with a multi-fractal cascade generator of fine timescale fluctuations, one can estimate several performance measures.

## 5 Conclusions and Further Work

In this paper we have investigated a structural modeling approach to describing the sub-frame or slice level traffic fluctuations in MPEG2 video traffic. This is a problem of engineering importance, as demonstrated by the performance impacts of these fine timescale fluctuations studied in this paper. The following summarizes our findings:

- the video trace shows the multi-scaling behavior over fine timescales (sub-frame or slice level - below 40 ms or so);
- queueing simulations show that these fine timescale fluctuations cause queueing backlogs even at low utilizations; specifically, ignoring the fine time fluctuations by assuming uniform cell generation below a frame level can cause significant underestimation of network delays;
- equally, "worst case assumptions" assuming that all the cells in the frame are delivered as a batch at the beginning of the frame can grossly overestimate network delays;
- the simulated performance obtained by "reconstructing" the finer timescale fluctuations using a multi-fractal cascade construction closely matches that obtained with the original trace, indicating that parsimonious models of the fine timescale fluctuations may be derived on the basis of multi-fractal cascades;
- we outline a conceptual approach to numerically analyze traffic generated by multi-fractal cascades, and demonstrate that a description consisting merely of multi-fractal *scaling exponents* is not complete;
- we indicate how a more complete description can be inferred from the long timescale behavior.

There are numerous avenues to expand this work, including repeating these experiments with additional video traces (in progress, some other data sets from the Atlanta Olympics [13] have been investigated, and show the behavior described in this paper); investigating finer timescale features at the block or macroblock level, if necessary; implementing the numerical methods to analyze traffic generated by multi-fractal cascades; and extensions to analyze and describe video traffic over the full range of engineering timescales of interest. In the longer term, the objective is to develop robust, tractable and parsimonious video traffic models and management methods that are usable in practice.

## Acknowledgments

We thank Walter Willinger for useful discussions, and Amarnath Mukherjee [13] for kindly supplying us with the data for this work. This work is supported in part by NSF grant NCR-9628067.

## References

1. P. Pancha and M. El-Zarki, "MPEG Coding for Variable Bit Rate Video Transmission", IEEE Communications, Vol. 32, No. 5, pp 54-66, May 1994.
2. D. P. Heyman and T.V. Lakshman, "Source models for VBR broadcast-video traffic", IEEE/ACM Trans. on Networking, Vol. 4, No. 1, pp 40-8, Feb 1996.
3. J. Beran, R. Sherman, M. S. Taqqu and W. Willinger, "Long-Range Dependence in Variable-Bit-Rate Video Traffic", IEEE Trans. Commun., Vol. 43, No. 2-4, pp. 1566-1579, Apr 1995.
4. A. Elwalid, D. Heyman, T. V. Lakshman and D. Mitra, "Fundamental bounds and approximations for ATM multiplexers with applications to video teleconferencing", IEEE J. Selected Areas in Comm., Vol. 13, No. 6, pp 1004-16, Aug. 1995.
5. W. E. Leland, M. S. Taqqu, W. Willinger, and D. V. Wilson, "On the Self-Similar Nature of Ethernet Traffic (Extended Version)", IEEE/ACM Trans. on Networking, Vol. 2, No. 1, pp. 1-15, Feb 1994.
6. A. Erramilli, O. Narayan, and W. Willinger, "Experimental Queueing Analysis with Long-Range Dependent Packet Traffic", IEEE/ACM Transactions on Networking, Vol. 4, No. 2, pp. 209-223, Apr 1996.
7. I. Norros, "A Storage Model with Self-Similar Input", Queueing Systems, Vol. 16, No. 3-4, pp. 387-396, 1994
8. A. Erramilli, O. Narayan, A. Neidhardt, I. Saniee, "Performance Impacts of Multi-Scaling in Wide Area TCP-IP Traffic", INFOCOM 2000 (to be published).
9. A. Gilbert, A. Friedman, W. Willinger, "Data Networks as cascades: Explaining the multifractal nature of Internet WAN traffic", Proc. ACM Sigcomm, 1998.
10. R.H. Reidy and J. Levy Vehel, "TCP Traffic is multifractal: A numerical study", INRIA Research Report No. 3129, Preprint 1997.
11. I. Saniee, "Multiscaling in Low-Aggregate Fast Packet Network Traffic", Bellcore Report, Sep 22, 1998.
12. The value of the transition point depends on the network; for instance, in Ref. [9], it is of the order of one second.
13. This data was supplied by Amarnath Mukherjee of Knoltex Corporation.
14. J.A. Bucklew, "Large deviation techniques in decision, simulation, and estimation" Wiley, New York, 1990.
15.  $\Lambda(s, t)/st$  is also referred to as the effective bandwidth for the traffic; see for instance F. Kelly, "Notes on effective bandwidths", in Stochastic Networks: Theory and Applications, ed. F.P. Kelly, S. Zachary and I. Ziedins, Oxford University Press, 1996.

# An Accurate Closed-Form Formula to Calculate the Dejittering Delay in Packetised Voice Transport

D. De Vleeschauwer<sup>1</sup>, G.H. Petit<sup>1</sup>, B. Steyaert<sup>2</sup>, S. Wittevrongel<sup>2</sup> and H. Bruneel<sup>2</sup>

<sup>1</sup> Alcatel Bell, Corporate Research Center,  
Francis Wellesplein 1, B-2018 Antwerpen, Belgium  
{danny.de\_vleeschauwer, guido.h.petit}@alcatel.be

<sup>2</sup> Ghent University, Vakgroep TELIN, SMACS Research Group,  
Sint-Pietersnieuwstraat 41, B-9000 Gent, Belgium  
{bs, sw, hb}@telin.rug.ac.be

**Abstract.** The transport of voice over a packet-based network introduces jitter in the voice flow. The jitter is compensated in the dejittering buffer by artificially delaying the first packet of the flow over the so-called dejittering delay and then cyclically reading the packets from the dejittering buffer. In this paper, we first give a general numerical method to compute the dejittering delay. Then, we introduce and justify a closed-form heuristic formula to calculate the dejittering delay for a network consisting of identically loaded, independent nodes modelled as M/G/1 queues. We demonstrate the accuracy of the heuristic formula by means of some numerical examples. Since the heuristic formula is closed-form, accurate, and explicitly shows the influence of certain network parameters, it is the preferred method to dimension the dejittering buffer and to perform sensitivity studies.

## 1 Introduction

Although the feasibility of packet-based transport of real-time voice has been proven a long time ago, a lot of interest has risen lately in specific implementations: e.g., Voice and Telephony over ATM (VTOA) [1], Voice over Frame Relay (VoFR) [2], and especially Voice over IP (VoIP) [3,4,5,6,7,8,9].

In the packetised voice transport the voice signal is partitioned in equally sized intervals. Each interval is encoded and the corresponding code word is transported in a packet over the packet-based network. Hence, the packets are produced with a constant interdeparture time equal to the length of an interval. Constant bit-rate codecs produce code words of constant size. Codecs with voice activity detection produce code words of constant size during talk spurts and produce (practically) no packets during periods of silence.

In the packet-based network the packets of several flows compete for the available resources. Some packets can be processed immediately when they arrive at a network node, while others have to wait. Some packets need to wait longer than others, causing so-called jitter in the voice flow. Because the decoder needs the packets at the original constant rate, the jitter is compensated in the dejittering buffer by artificially delaying the first packet of a flow over the so-called dejittering delay and then

cyclically reading the packets from the dejittering buffer [8,9]. The dejittering delay has to be carefully chosen. A too large a choice jeopardises the real-time delivery of the voice. A too small a choice results in a lot of packets arriving in the dejittering buffer after they were supposed to be read, which means that they are effectively lost.

In this paper we develop a heuristic formula to calculate the dejittering delay to compensate for the jitter introduced in a network dedicated to the transport of voice, i.e., free of data. For voice transported over a packet-based network simultaneously carrying data, the heuristic formula is of some use too (see [8,9]).

In the following sections, the network nodes traversed by a given voice flow will be modelled as a sequence of independent M/G/1 queues. This choice can be motivated as follows. First, note that for a given voice flow, packets enter the network at a constant rate (possibly with some gaps in between in case of voice activity detection), which implies that modelling these voice packet arrival streams at the network's ingress by a Poisson process is actually a worst-case scenario. In [10], it has now been argued that if a number of flows that are 'better than Poisson' (i.e., with lower variability of the interarrival times) are multiplexed in a FIFO buffer, then the outgoing streams continue to be better than Poisson. Thus it is clear that this property will be propagated throughout the subsequent network nodes visited by a flow, provided that these nodes can be regarded as being independent. This will be the case if a given flow accounts only for a small fraction of the total amount of traffic in a node (see e.g. [11]) which is the case in a (dedicated) packet-based VoIP network, where a large number of low bit rate voice flows are to be multiplexed in each node.

In the next section, the mathematical model to determine the dejittering delay for a dedicated packet-based voice network is described. Section 3 introduces and justifies a closed-form heuristic formula to calculate the dejittering delay. In section 4, the performance of this heuristic formula is demonstrated with some numerical results. In the last section some conclusions are drawn.

## 2 The Dejittering Delay for a Network of M/G/1 Queues

In order to transport interactive voice over a packet-based network, the voice signal is first encoded and the corresponding code words are transported in packets of constant size. Such a flow of packets carrying voice requires real-time delivery; i.e., the packets have to arrive at the destination within a certain delay bound. In each node of the network the packets of the flow experience a delay, consisting of a deterministic service time (determined by the packet size) and a stochastic waiting time. The stochastic waiting time introduces jitter in the voice flow. Since the voice decoder requires the packets at a constant rate, this jitter has to be compensated.

In a network of M/G/1 queues the largest possible total waiting time is infinite. When the  $(1-P)$ -quantile  $w_p$  is used as dejittering delay a fraction  $P$  of the packets are lost in the dejittering buffer. Fortunately, voice codecs can tolerate some packet loss. Depending on the codec a packet loss in the range  $[10^{-5}, 10^{-2}]$  can be tolerated.

Consider a route through a dedicated VoIP network (i.e., a network free of data) traversing  $N$  identically loaded, independent nodes modelled as M/G/1 queues. The service discipline in each node is FIFO, because all packets in the network require real-time delivery, and hence, have the same priority. The probability density function

(pdf)  $p_W(w)$  of the waiting time  $W$  introduced in the network is known under the form of the Laplace transform

$$W(s) = E\{e^{-sW}\} = \int dx p_W(x) e^{-sx} . \quad (1)$$

We use the following definitions. The load of the voice traffic is  $\rho$ . The Laplace transform of the pdf of the service time required by a voice packet is  $B(s)$ . The  $k$ -th moment of this service time is denoted as  $b_k$ . The Laplace transform of the pdf of the residual service time, which is the remaining service time of the voice packet in service observed by an arriving voice packet, is

$$R(s) = \frac{1 - B(s)}{b_1 s} . \quad (2)$$

With these definitions, the Laplace transform of the waiting time pdf introduced in a single node, modelled as an M/G/1 queue, is given by [12]

$$W_1(s) = \frac{1 - \rho}{1 - \rho R(s)} . \quad (3)$$

In the introduction, it was argued that for the case of a network consisting of multiple identical nodes, the waiting times encountered in consecutive nodes can be regarded as being statistically independent. Hence, if we denote by  $W_N$  the total waiting time encountered through  $N$  nodes, its pdf is obtained from the  $N$ -fold convolution of the individual waiting time's pdf, which yields for the corresponding Laplace transform

$$W_N(s) = [W_1(s)]^N . \quad (4)$$

The tail distribution of the total waiting time  $W_N$  is defined as

$$\Pr[W_N > w] = t_W(w) = \int_w^\infty dx p_W(x) , \quad (5)$$

where  $p_W(x)$  is the inverse of the Laplace transform (4). This transform can be inverted numerically by using a Fast Fourier Transform (FFT). The major disadvantages of this approach are that it requires a lot of computation time and that it becomes error-prone for low values of  $P$ . If the calculations are performed in single precision, the influence of the rounding errors introduced in this procedure is already noticeable for  $P$ -values around  $10^{-4}$ . For smaller  $P$ -values rounding errors have too large an influence to lead to meaningful results. Therefore, we propose the analytic/heuristic approach described in the following sections. Our objective is to derive a closed-form formula for the  $(1-P)$ -quantile  $w_p$  of the total waiting time from transform (4).

### 3 A Heuristic Formula

In this section we introduce (see [13]) and justify a heuristic formula, to determine the  $(1-P)$ -quantile  $w_p$  of the total waiting time. The heuristic formula states that the  $(1-P)$ -quantile is equal to the sum of the average total waiting time  $\mu_N$  and a number of times the standard deviation  $\sigma_N$  of the total waiting time:

$$w_p = \mu_N + \alpha_N(P)\sigma_N \quad (6)$$

The value of  $\alpha_N(P)$  depends solely on the number  $N$  of nodes traversed and the value of  $P$ , but is independent of the service time characteristics of a voice packet. The factor  $\alpha_N(P)$  is chosen such that if the waiting time in 1 node were exponentially distributed, the heuristic formula (6) would be exact. It follows that

$$\alpha_N(P) = \frac{E_N^{-1}(P) - N}{\sqrt{N}} \quad (7)$$

where

$$E_N(w) = \int_w^\infty dx \left[ \frac{x^{N-1}}{(N-1)!} \exp(-x) \right] = \exp(-w) \sum_{j=0}^{N-1} \frac{w^j}{j!} \quad (8)$$

is the tail distribution of the normalised Erlang distribution of degree  $N$ , which is equal to the  $\chi^2$ -distribution with  $2N$  degrees of freedom. This latter function and its inverse are included in standard spreadsheets, e.g. Excel. Hence, formula (6) is closed-form. The values  $\alpha_N(P)$  are tabulated in Table 1. In the next subsections, we justify the heuristic formula (6) and determine its range of applicability.

**Table 1.** The weighting factor  $\alpha_N(P)$ .

$\alpha_N(P)$	$N=1$	$N=2$	$N=4$	$N=8$	$N=16$	$N=32$	$N=2^7$	$N=2^{10}$	$N=2^{13}$	$N \rightarrow \infty$
$P=10^{-2}$	3.605	3.280	3.023	2.828	2.686	2.582	2.455	2.372	2.342	2.327
$P=10^{-3}$	5.908	5.115	4.531	4.111	3.811	3.599	3.344	3.180	3.122	3.091
$P=10^{-4}$	8.210	6.899	5.957	5.290	4.821	4.493	4.102	3.853	3.766	3.719
$P=10^{-5}$	10.513	8.653	7.333	6.407	5.762	5.312	4.781	4.445	4.328	4.265
$P=10^{-6}$	12.816	10.386	8.675	7.482	6.654	6.080	5.404	4.980	4.833	4.753
$P=10^{-7}$	15.118	12.106	9.993	8.526	7.511	6.809	5.987	5.473	5.296	5.199
$P=10^{-8}$	17.421	13.814	11.292	9.545	8.340	7.509	6.538	5.933	5.725	5.612
$P=10^{-9}$	19.723	15.514	12.577	10.546	9.147	8.185	7.063	6.367	6.127	5.997
$P=10^{-10}$	22.026	17.207	13.850	11.530	9.936	8.840	7.566	6.778	6.507	6.361

#### 3.1 Asymptotic-Tail Approximation

For practical Laplace transforms  $B(s)$  of the service time pdf,  $W_1(s)$  has a clear dominant pole  $p$ , i.e., a real pole with multiplicity 1 and an absolute value (much)



smaller than the absolute values of the other poles. This implies that eq. (3) can be written as

$$W_1(s) = \frac{F(s)}{(1 + \lambda s)} \quad , \quad (9)$$

with  $\lambda = -1/p$  and  $F(s)$  an analytical function in the neighbourhood of the dominant pole  $p$ . Due to eq. (3), the pole  $p$  satisfies  $1 - \rho R(p) = 0$ . We refer to  $\lambda$  as the rate of decay of the tail.

By using the residue theorem [12], we can approximate the pdf of the waiting time in a network consisting of 1 node as

$$p_W(x) \approx \lim_{s \rightarrow p} \left[ \frac{1}{\lambda} F(s) \exp(sx) \right] = \frac{1}{\lambda} \exp\left(-\frac{x + \delta}{\lambda}\right) \quad (10)$$

with

$$\delta = \frac{\ln(F(p))}{p} \quad . \quad (11)$$

Eq. (10) is asymptotically exact, i.e., for sufficiently large values of  $x$ . Hence, the  $(1-P)$ -quantile of the waiting time in 1 node is given by

$$w_P = \lambda E_1^{-1}(P) - \delta \quad . \quad (12)$$

For a network consisting of  $N$  nodes, the multiplicity of the dominant pole  $p$  is  $N$ . In this case, use of the residue theorem [12] leads to the following approximation for the pdf of the total waiting time:

$$p_W(x) \approx \lim_{s \rightarrow p} \left\{ \frac{1}{(N-1)!} \frac{d^{N-1}}{ds^{N-1}} \left[ \left( \frac{1}{\lambda} \right)^N (F(s))^N \exp(sx) \right] \right\} \quad . \quad (13)$$

Because of the derivative appearing in eq. (13), the case of  $N$  nodes is considerably more difficult to handle than the case of 1 node. Inspired by eq. (11) we now state that

$$F(s) = \exp(\delta s) \quad (14)$$

in the neighbourhood of the dominant pole  $p$ . As seen above, this assumption leads to the correct result for the case  $N=1$ . Intuitively, it is expected that for low values of  $N$ , this approximation will be reasonably good, but when  $N$  increases the errors introduced by this approximation will become larger. With the approximation (14) the calculation of the residue of eq. (13) is trivial, and hence, the tail distribution can easily be calculated as

$$t_W(w) = E_N \left( \frac{w + N\delta}{\lambda} \right) \quad . \quad (15)$$

The  $(1-P)$ -quantile is then given by

$$w_p = \lambda E_N^{-1}(P) - N\delta \quad , \quad (16)$$

which is the multi-node extension of eq. (12). We refer to eq. (16) as ‘the formula based on the method of the dominant pole’, meaning that we use the exact (numerically calculated) values of  $\lambda$  and  $\delta$ . It is only a good approximation if there is a clear dominant pole and if the number of nodes in the network  $N$  is low. Since eq. (16) requires the exact knowledge of the dominant pole  $p$ , which involves a root finding method to calculate  $p$  numerically, this formula is not closed-form. Therefore, we attempt to express  $p$  in terms of known quantities.

### 3.2 Heavy-Load Approximation

In order to obtain a closed-form formula, we introduce a heavy-load approximation for  $\lambda$  and  $\delta$ . This is done as follows. Under heavy load  $\rho$ , the Laplace transform of eq. (3) can be approximated by

$$W_1(s) \approx \frac{1}{1 + As - \tau \frac{(As)^2}{2}} \quad , \quad (17)$$

with

$$A = \frac{\rho}{1-\rho} \frac{b_2}{2b_1} \quad ; \quad \tau = \frac{\rho}{1-\rho} \frac{b_3}{3b_1} \frac{1}{A^2} \quad . \quad (18)$$

Notice that as  $\rho$  tends to 1,  $\tau$  tends to 0. Therefore, in the following, we neglect terms of the order of  $\tau^2$ . Also note that the parameters  $A$  and  $\tau$  are closely related to the average  $\mu_1$  and the standard deviation  $\sigma_1$  of the waiting time in a single node:

$$\mu_1 = A \quad ; \quad \sigma_1 = A\sqrt{1+\tau} \approx A\left(1 + \frac{\tau}{2}\right) \quad . \quad (19)$$

With the approximation (17) for  $W_1(s)$  the rate of decay of the tail is given by

$$\lambda = \frac{A\tau}{\sqrt{1+2\tau}-1} \approx A\left(1 + \frac{\tau}{2}\right) \approx \sigma_1 \quad . \quad (20)$$

This states that the rate of decay of the tail  $\lambda$  can be very well approximated by the standard deviation  $\sigma_1$  of the waiting time in 1 node.

Furthermore, (still under approximation (17)) we readily see from

$$\lim_{s \rightarrow p} (1 + \lambda s) W_1(s) = \frac{\tau}{1 + 2\tau - \sqrt{1 + 2\tau}} \quad , \quad (21)$$

that

$$\delta = \lambda \ln \left( \frac{1 + 2\tau - \sqrt{1 + 2\tau}}{\tau} \right) \approx \frac{\lambda\tau}{2} \approx \sigma_1 - \mu_1 \quad . \quad (22)$$

With the approximations (20) and (22) for  $\lambda$  and  $\delta$ , eq. (16) can easily be rewritten in the form of the heuristic formula of eq. (6).

### 3.3 Large-Network Approximation

If the number of nodes traversed is large, the approximation of eq. (14) no longer holds and the formula based on the method of the dominant pole (eq. (16)) becomes inaccurate. In that case, however, the law of large numbers justifies the heuristic formula (6). When a large number  $N$  of nodes is traversed the pdf of the total waiting time tends to a Gaussian with average  $\mu_N$  and standard deviation  $\sigma_N$ . In this case, the tail distribution is approximated by

$$t_W(w) = \text{erfc} \left( \frac{w - \mu_N}{\sigma_N} \right) = \frac{1}{\sqrt{2\pi}} \int_{\frac{w - \mu_N}{\sigma_N}}^{\infty} dx \exp \left( -\frac{x^2}{2} \right) \quad . \quad (23)$$

Hence, the  $(1-P)$ -quantile is given by

$$w_P = \mu_N + \text{erfc}^{-1}(P)\sigma_N \quad . \quad (24)$$

Since the normalised Erlang distribution of degree  $N$  (see eq. (8)) converges to a Gaussian distribution with average value and variance both equal to  $N$  as  $N$  tends to infinity, we can immediately conclude that (see eq. (7))

$$\lim_{N \rightarrow \infty} \alpha_N(P) = \text{erfc}^{-1}(P) \quad . \quad (25)$$

This behaviour can also be observed in Table 1. Hence, eq. (24) is clearly of the form of the heuristic formula (6).

## 4 Numerical Results

In this section the performance of the heuristic formula is demonstrated with some numerical examples. We consider the M/D/1 queue and M/D<sub>1</sub>+D<sub>2</sub>/1 queue.

The M/D/1 queue may be used for an IP network transporting voice where all voice codecs produce flows of packets of the same size. In this case, all voice packets have the same deterministic service time, and the corresponding Laplace transform is given by

$$B(s) = \exp(-s) \quad . \quad (26)$$

Remark that the average service time is normalised, i.e.  $b_1=1$ . When  $N$  nodes are traversed, the average  $\mu_N$  and standard deviation  $\sigma_N$  of the total waiting time are:

$$\mu_N = N \left( \frac{\rho}{2(1-\rho)} \right) ; \quad (27)$$

$$\sigma_N = \sqrt{N} \sqrt{\left( \frac{\rho}{2(1-\rho)} \right)^2 + \frac{\rho}{3(1-\rho)}} . \quad (28)$$

The  $M/D_1+D_2/1$  queue may e.g. be used for an IP network transporting voice, where the voice flows are produced by one of two types of codecs: e.g. a codec producing 64 kb/s and a low bit rate codec producing 8 kb/s. Because in delay sensitive services (e.g. voice services) the end-to-end delay is limited (to e.g. 150 ms), the packetisation delay, i.e., the time to fill a packet, is limited too. Therefore, the packets for the 64 kb/s codec are likely to be larger than the packets for the 8 kb/s codec. Because on the one hand also queueing delay and dejittering delay and on the other hand also the header size play a role, the packet size ratio will not be exactly 8 to 1, but slightly less. Here, we take a ratio of 7 to 1 as an example. Furthermore, we assume that half of the flows use the 8 kb/s codec, the other half use the 64 kb/s codec. Hence, the Laplace transform of the pdf of the service time is

$$B(s) = 0.5 \exp(-0.25s) + 0.5 \exp(-1.75s) . \quad (29)$$

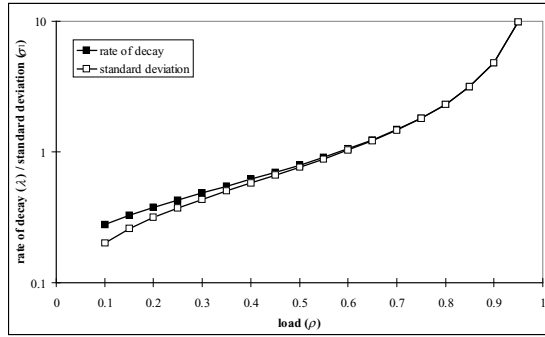
Remark that again the average service time is normalised, i.e.  $b_1=1$ . When  $N$  nodes are traversed, the average  $\mu_N$  and standard deviation  $\sigma_N$  of the total waiting time are:

$$\mu_N = N \left( \frac{\rho}{2(1-\rho)} 1.5625 \right) ; \quad (30)$$

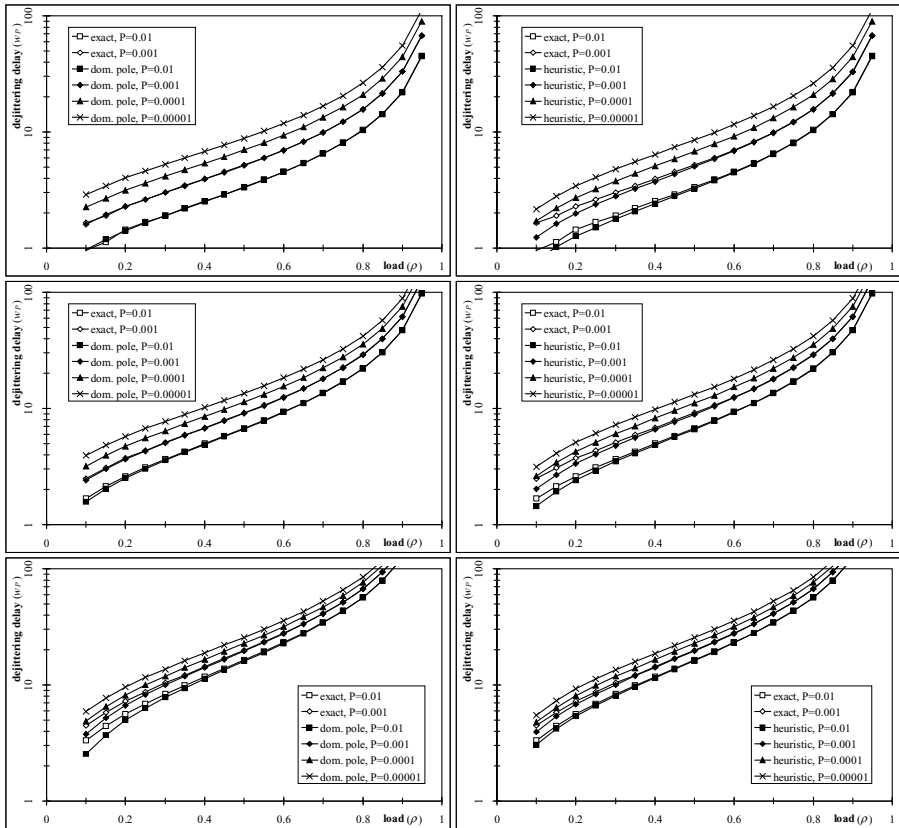
$$\sigma_N = \sqrt{N} \sqrt{\left( \frac{\rho}{2(1-\rho)} 1.5625 \right)^2 + \frac{\rho}{3(1-\rho)} 2.6875} . \quad (31)$$

The accuracy of the approximation for the rate of decay of the tail  $\lambda$  introduced in eq. (20) is illustrated in Figure 1 and Figure 3. These figures show that for sufficiently large loads the rate of decay of the tail  $\lambda$  is very well approximated by the standard deviation  $\sigma_1$  of the waiting time in 1 node.

Figure 2 and Figure 4 show how the dejittering delay increases as the load  $\rho$  increases for a flow traversing  $N$  nodes modelled as  $M/D/1$  queues and  $M/D_1+D_2/1$  queues respectively. The dejittering delay associated with several  $P$ -values is calculated with the method of the dominant pole (eq. (16)) and with the heuristic formula (eq. (6)). For  $P=10^{-2}$  and  $P=10^{-3}$  the exact dejittering delay (obtained via the FFT) is also given. For smaller values of  $P$ , the FFT-based method becomes numerically unstable, and the use of an efficient alternative method to calculate the dejittering delay, such as the one introduced in this paper, becomes mandatory. It can be seen that both methods produce accurate results for all values of  $N$  and  $P$ , if the



**Fig. 2.** The rate of decay of the tail ( $\lambda$ ) and the standard deviation of the waiting time ( $\sigma_1$ ) for the M/D/1 queue.



**Fig. 2.** The dejittering delay associated with several values of the packet loss  $P$  for a network consisting of  $N$  M/D/1 nodes. The number of nodes  $N$  is 1 (top), 4 (middle), or 16 (bottom). The method of the dominant pole (left) is compared with the heuristic formula (right). The exact solution (obtained via the FFT) is also given for large  $P$ -values (0.01 and 0.001).

load  $\rho$  is above 0.5. For networks consisting of a moderate amount of nodes ( $N \leq 4$ ), the method of the dominant pole slightly outperforms the heuristic formula. For large networks the heuristic formula is slightly better. This behaviour can be explained as follows. For a network consisting of 1 node the only approximation the method of the dominant pole makes is that it neglects the influence of the poles other than the dominant pole. The heuristic formula further introduces approximation (20), i.e., it does not use the exact rate of decay of the tail, but approximates it by the standard deviation of the waiting time in 1 node. Hence, in a network consisting of 1 node the heuristic formula is outperformed by the method of the dominant pole. For a large network, on the other hand, the method of the dominant pole also has to introduce approximation (14). Again the heuristic formula further introduces approximation (20). Apparently, both approximations (14) and (20) roughly compensate each other, if the number of network nodes becomes large. This compensation effect is justified by the large-network approximation of Section 3.3. Hence, for the case of a large network the heuristic formula becomes more accurate than the method of the dominant pole.

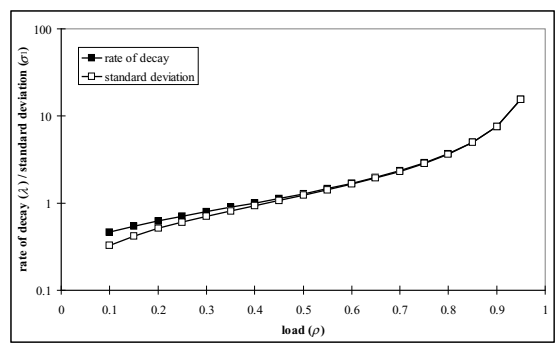
Remark by comparing Figure 4 with Figure 2 that the dejittering delay for a network of  $M/D_1+D_2/1$  nodes is about 60% higher than the dejittering delay for a network of  $M/D/1$  nodes. With the heuristic formula this can be readily concluded by comparing the formulae for the average delay and standard deviation of the delay (eq. (27) with eq. (30) and eq. (28) with eq. (31)). To be able to draw the same conclusion with the FFT-based method or the method of the dominant pole the curves of Figure 2 and Figure 4 have to be explicitly computed.

## 5 Discussion and Conclusions

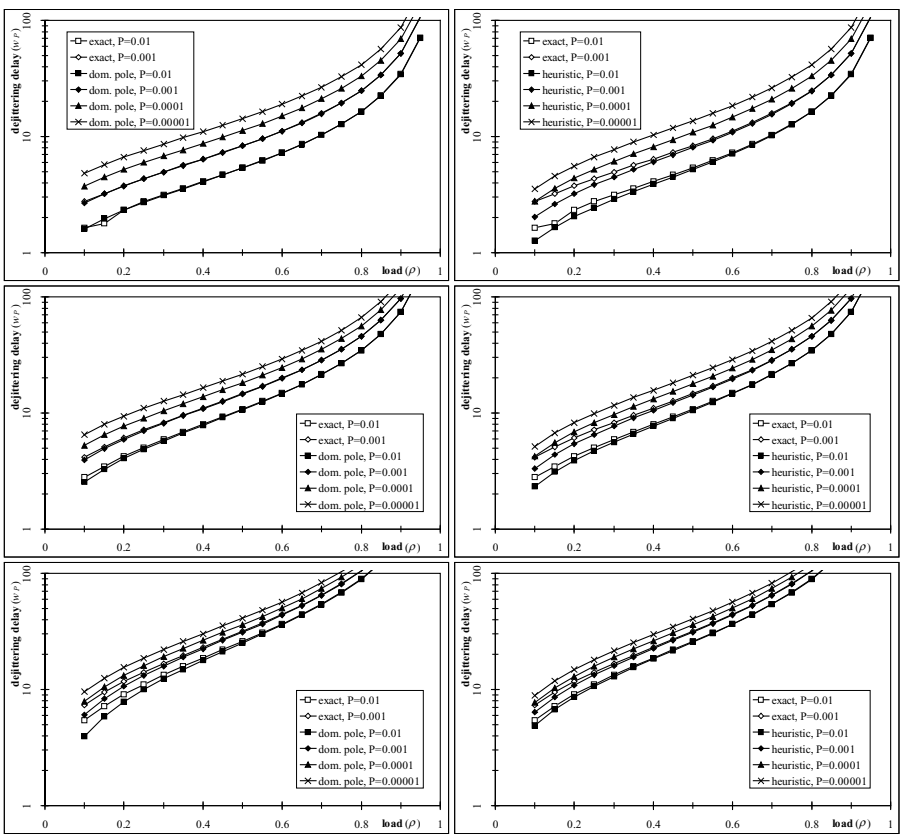
With the theoretical justification of Section 3 and the numerical examples of Section 4, we conclude that the heuristic formula (6)

1. explicitly shows the influence of certain parameters: the dejittering delay is the sum of the average total waiting time and a number of times the standard deviation of the total waiting time; the weight of the standard deviation only depends on the number of nodes traversed and the fraction of packets that (are allowed to) arrive too late in the dejittering buffer;
2. slightly underestimates the dejittering delay at low loads;
3. produces very accurate results for all values of  $N$  and  $P$  when the load is larger than 0.5; and
4. is slightly outperformed by the method of the dominant pole (16) for a network consisting of a moderate amount of nodes, but outperforms the method of the dominant pole for large networks.

Furthermore, having a closed-form formula for the dejittering delay has the advantage that it explicitly shows how the dejittering delay increases as the variability of the service time increases ( $b_2$  and  $b_3$  increase while keeping  $b_1=1$ ). Also we readily observe that the dejittering delay for a network consisting of  $N$  nodes is much smaller than  $N$  times the dejittering delay for 1 node. Estimating the dejittering delay for a network consisting of  $N$  nodes by taking  $N$  times the dejittering delay for 1 node leads to a large overestimation of the dejittering delay.



**Fig. 4.** The rate of decay of the tail ( $\lambda$ ) and the standard deviation of the waiting time ( $\sigma_1$ ) for the  $M/D_1+D_2/1$  queue.



**Fig. 4.** The dejittering delay associated with several values of the packet loss  $P$  for a network consisting of  $N = M/D_1+D_2/1$  nodes. The number of nodes  $N$  is 1 (top), 4 (middle) or 16 (bottom). The method of the dominant pole (left) is compared with the heuristic formula (right). The exact solution (obtained via the FFT) is also given for large  $P$ -values (0.01 and 0.001).

## Acknowledgements

This work was carried out within the framework of the project LIMSON sponsored by the Flemish Institute for the Promotion of Scientific and Technological Research in the Industry (IWT). The fourth author is a postdoctoral fellow of the Fund for Scientific Research – Flanders (Belgium) (F.W.O.).

## References

1. D.J. Wright, "Voice over ATM: An Evaluation of Implementation Alternatives", IEEE Communications Magazine, Vol. 34, No. 5, pp. 72-80, May 1996.
2. C. Bucholtz, "Another Avenue for Voice", Telephony, pp. 60, May 1997.
3. A. Cray, "Voice over IP, Hear's how", Data Communications, pp. 44-58, Apr. 1998.
4. R. Gareiss, "Voice over IP Services: The Sound Decision", Data Communications, pp. 75-84, Mar. 1998.
5. T.J. Kostas, M.S. Borella, I. Sidhu, G.M. Schuster, J. Grabiec, J. Mahler, "Real-Time Voice over Packet-Switched Networks", IEEE Network, pp. 18-27, Jan./Feb. 1998.
6. T. Lewis, "VoIP; Killer App for the Internet", IEEE Internet Computing, pp. 110-112, Nov.-Dec. 1997.
7. S.R. Ahuja, K.G. Murti, "Packet Telephony", Bell Labs Technical Journal, pp. 5-14, Spring 1997.
8. K. Van Der Wal, M. Mandjes, H. Bastiaansen, "Delay Performance Analysis of the New Internet Services with Guaranteed QoS", Proceedings of the IEEE, Vol. 85, No. 12, pp. 1947-1957, Dec. 1997.
9. M. Mandjes, K. Van Der Wal, R. Kooij, H. Bastiaansen, "End-to-end Delay Models for Interactive Services on Large Scale IP Networks", Proceedings of IFIP'99 (Antwerp, Belgium, 28-30 June 1999).
10. F. Bricchet, L. Massoulié, J.W. Roberts, "Stochastic Ordering and the Notion of Negligible CDV", Proceedings of ITC'15 (Washington, June 1997), pp. 1433-1444.
11. W.-C. Lau, S.-Q. Li, "Traffic Distortion and Inter-source Cross-correlation in High-speed Integrated Networks", Computer Networks and ISDN Systems, Vol. 29, pp. 811-830, 1997.
12. L. Kleinrock, "Queueing Systems, Vol. 1: Theory", John Wiley and Sons, New York, 1975.
13. G.H. Petit, "A BASIC Tool to Calculate Jitter", BASIC program, Personal communication, 1988.



# Interworking of B-ISDN Signaling and Internet Protocol

Muneyoshi Suzuki

NTT Information Sharing Platform Laboratories  
3-9-11, Midori-cho, Musashino-shi, Tokyo 180-8585, Japan  
suzuki@nal.ecl.net

**Abstract.** In the ITU-T and IETF, I proposed the assignment of the information field and protocol identifier in the Q.2941 GIT (Generic Identifier Transport) and Q.2957 UUS (User-to-User Signalling) for the Internet protocol. The aim is to enable B-ISDN signaling support to be used for session of the Internet protocol. The purpose of this paper is to clarify how these specifications enable B-ISDN signaling support to be used for long-lived and QoS-sensitive sessions.

Long-lived session requires that the B-ISDN signaling be capable of transferring a session identifier, which is currently supported only by proprietary protocols. By using the GIT information element, B-ISDN signaling can be used to support it.

To implement ATM VC support for QoS-sensitive session, this paper describes three approaches. In QoS-sensitive session using these approaches, the B-ISDN signaling must transfer the session identifier or IP signaling protocol. Therefore, B-ISDN signaling based on current recommendations and standards cannot support QoS-sensitive session. These problems can be solved by using the GIT or UUS information element, as I have proposed.

## 1 Introduction

With the development of new multimedia applications for the Internet, the need for multimedia support in the IP network, which currently supports only best-effort communications, is increasing. In particular, QoS-guaranteed communications [1,2] is needed to support the voice, audio, and video communications applications being developed. Mechanisms will also be needed that can efficiently transfer the huge volume of traffic expected with these applications.

The major features of B-ISDN are high speed, logical multiplexing using VPs and VCs, and flexible QoS management per VC, so it is quite natural to use these distinctive functions of B-ISDN to implement a multimedia-support mechanism in the Internet. When a long-lived session<sup>1</sup> is supported by a particular VC, efficient packet forwarding is possible by using the high-speed and

---

<sup>1</sup> The Internet protocol reference model does not contain the session and presentation layers, so communication between end-to-end applications over the Internet is equivalent to a session; the term "session" is thus used in this paper.

logical multiplexing of B-ISDN [3,4]. The flexible QoS management and logical multiplexing functions in B-ISDN will enable QoS-guaranteed communications to be implemented on the Internet [5,6].

Currently on the Internet, N-ISDN is widely used as a datalink media. In the B-ISDN technology area, development of classical IP over ATM [7] and LAN emulation [8] technologies, which use a VC as the datalink media, has been completed. These technologies are already implemented in a large number of products. However, the development of Internet technologies that use the distinctive B-ISDN features described in above is progressing slowly. There are a number of reasons for this. One is that for applications to be able to practically use B-ISDN features, advanced functions not supported by N-ISDN must be used. However, the architecture of B-ISDN signaling is basically the same as that of N-ISDN signaling, so it does not meet the requirements of such applications.

Therefore, in the ITU-T and IETF, I proposed the assignment of the information field and protocol identifier in the Q.2941 GIT (Generic Identifier Transport) [9] and Q.2957 UUS (User-to-User Signalling) [10] for the Internet protocol [11,12,13]. The aim is to enable B-ISDN signaling support to be used for session of the Internet protocol; ITU-T recommendations and a standard track RFC based on this proposal will be published. The purpose of this paper is to clarify how these specifications enable B-ISDN signaling support to be used for long-lived and QoS-sensitive sessions.

First, this paper describes a method for implementing ATM VC support for long-lived session and explains why B-ISDN signaling must be capable of transferring a session identifier. Then it explains why the current B-ISDN signaling cannot satisfy this requirement and shows how the problem can be solved by using the GIT information element.

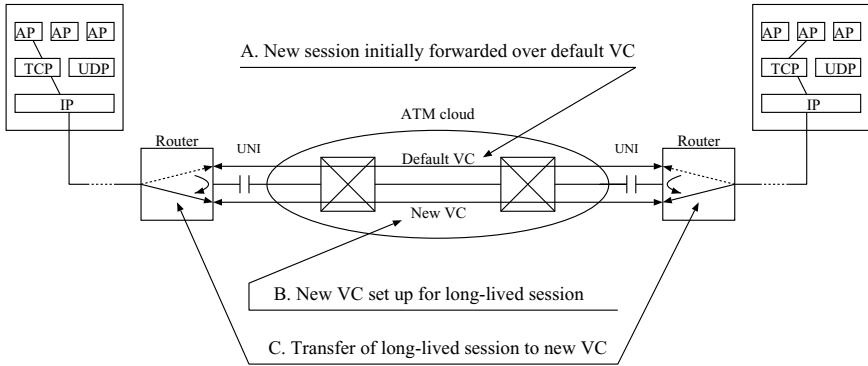
Next, this paper describes three approaches to implementing ATM VC support for QoS-sensitive session and compares the features and problems of these approaches. With these approaches, the B-ISDN signaling must be capable of transferring the session identifier or IP signaling protocol. Then it explains why the current B-ISDN signaling cannot satisfy this requirement and explains how the problem can be solved by using the GIT or UUS information element.

## 2 Long-Lived Session Signaling

### 2.1 ATM VC Support for Long-Lived Session

An example scenario of ATM SVC support for a long-lived session is shown in Fig. 1. First, a session is multiplexed into the default VC connecting the routers. Then, if a router detects that it is a long-lived session, the router sets up a new VC for the session. After the new VC is established, the session is moved to it [3,4].

ATM PVC support for long-lived session can be implemented as follows: PVCs connecting the routers are pre-configured, and when a router detects a long-lived session, it selects an available PVC and moves the long-lived session to it.



**Fig. 1.** Example scenario of ATM SVC support for long-lived session

ATM VC support for long-lived session was proposed by Newman and Katsube [3,4]. Support is provided by using the high-speed and logical multiplexing features of B-ISDN, which enable efficient packet forwarding. However, these proposals do not use the standard B-ISDN signaling protocol; proprietary protocols are used instead.

## 2.2 Requirements for B-ISDN Signaling

If the ATM SVC support for long-lived session described above is implemented using B-ISDN signaling, the B-ISDN signaling entity in the called-side router must detect that the incoming call corresponds to a session of the Internet protocol and notify the IP-layer entity. Based on this information, the IP-layer entity would move the session to the new VC. Therefore, to implement this signaling procedure, the SETUP message in the B-ISDN signaling must include a session identifier as an information element.

A session in the Internet is identified by a combination of the source and destination IP addresses, the protocol number, and the source and destination port numbers. The length of an IPv4 address is 4 octets, and that of an IPv6 address is 16 octets; the protocol number takes 1 octet, and the port number takes 2 octets, so the length of the session identifier for IPv4 is 13 octets and for IPv6 is 37 octets [14,15,16,17]. A session of the ST2+ (IPv5) is identified by the SID, which is 6 octets long [1].

Therefore, to enable ATM SVC support for long-lived session, the SETUP message in the B-ISDN signaling must be capable of transferring a session identifier of at least 37 octets long.

## 2.3 Problems with Current B-ISDN Signaling and Solution

**Session identifier in Q.2931 and UNI 3.1.** The SETUP message defined in the Q.2931 and UNI 3.1 signaling protocol specifications contains a B-HLI

(broadband high-layer information) information element that could be used to transfer the session identifier [18,19]. However, when “Vendor Specific” is selected for the high layer information type in the B-HLI element, the identifier is restricted to 4 octets, and when “User Specific” is selected, it is restricted to 8 octets. Therefore, the SETUP message defined in the Q.2931 and UNI 3.1 specifications cannot hold the session identifier.

**Session identifier in SCS2 and SIG 4.0.** The SETUP message defined in the SCS2<sup>2</sup> and SIG 4.0 signaling protocol specifications contains B-HLI and GIT information elements that could be used to transfer the session identifier [9,20]. The GIT element enables the transfer of identifiers between end-to-end users in an ATM network and is an optional information element for the UNI signaling protocol. In the SCS2 and SIG 4.0 specifications, signaling messages transferred between end-to-end users may contain up to three GIT information elements, and the ATM network transfers the elements transparently.

However, in these specifications, the maximum length of the GIT element is 33 octets, so it cannot hold the IPv6 session identifier. Furthermore, they also do not assign an identifier type for the Internet protocol.

**Solution using GIT.** To solve these problems, I proposed the assignment of the information field and protocol identifier in the Q.2941 GIT for the Internet protocol. It is also proposed increasing the maximum length of the information element.

A GIT information element consists of a common header and identifiers. Each identifier consists of identifier type, length, and identifier value fields. The sum of the length of the common header, identifier type, and length fields is 7 octets. The maximum length of the session identifier for the Internet protocol is 37 octets, so the length of the GIT information element must be extended to at least 44 octets. In the new GIT recommendation, the length is extended to 63 octets, and identifier types for the IPv4, ST2+, IPv6, and MPLS protocols are supported.

### 3 QoS-Sensitive Session Signaling

#### 3.1 ATM VC Support for QoS-Sensitive Session

The major difference between ATM VC support for long-lived and QoS-sensitive sessions is whether SVC setup or PVC selection timing is used. In the former, a setup or selection is initiated when a long-lived session is detected, but in the latter, it is initiated when resource is reserved by the IP signaling protocol such as the ST2+ and RSVP. In the latter case, the ATM network between routers must forward the IP signaling protocol.

<sup>2</sup> Signalling Capability Set No. 2: series of B-ISDN signaling recommendations issued by the ITU-T.

There has been little discussion of connection-oriented network support for connection-oriented protocols. Damaskos and Gavras investigated B-ISDN signaling support for the BERKOM Transport System [21], and N-ISDN signaling support for OSI connection-mode network service is mentioned in ITU-T Recommendation Q.923 [22]. However, implementation of signaling protocol support for connection-oriented protocols was not addressed completely.

Damaskos and Gavras discussed two schemes for forwarding the IP signaling protocol over an ATM network. One is to multiplex the protocol into the default VC connecting the routers or to forward the protocol through a particular VC. This scheme is called the sequential approach; resource reservation in the IP layer and SVC setup are initiated sequentially. The second scheme is to forward the IP signaling protocol as an information element in the B-ISDN signaling. This scheme is called the simultaneous approach; resource reservation in the IP layer and SVC setup are initiated simultaneously [21].

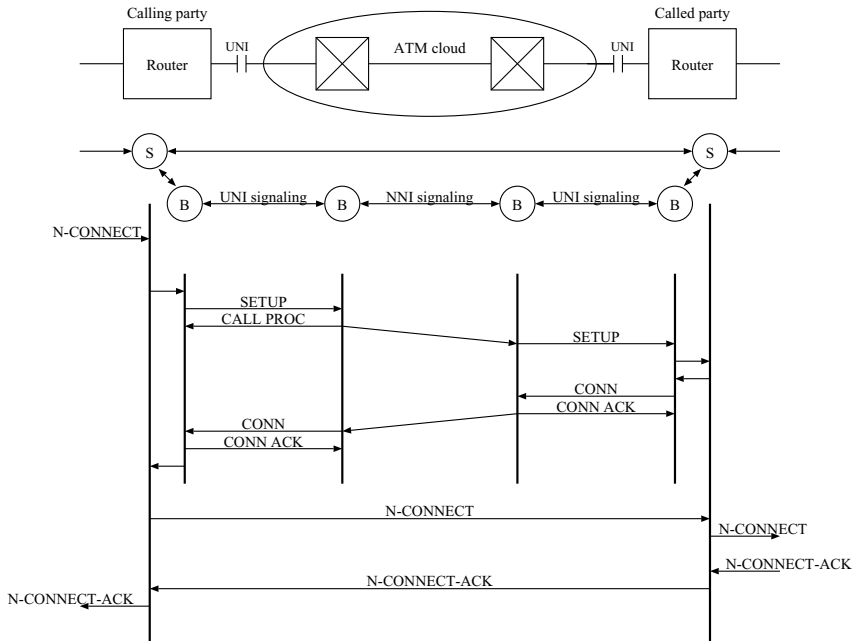
The sequential approach can be further classified into two practical schemes. One is to initiate resource reservation in the IP layer after SVC establishment; it is called the bottom-up approach in this paper. The second is to initiate SVC establishment after resource reservation; it is called the top-down approach in this paper. These sequential approaches are applicable if the ATM network supports only PVC; the simultaneous approach is not.

### 3.2 Comparison of IP Signaling Protocol Forwarding Schemes

This section compares the features and problems of these three approaches for forwarding the IP signaling protocol. The following figures show example procedures for enabling ATM SVC support for QoS-sensitive session based on IP and B-ISDN signaling protocols across the UNIs in the ATM network connecting the routers. In the figures, an “S” means a IP signaling entity and a “B” means a B-ISDN signaling entity. These figures do not show the VC setup procedure that forwards the B-ISDN or IP signaling protocol.

Both ST2+ and RSVP have been proposed for the IP signaling protocol, and other IP signaling protocols are likely to be developed in the future. Therefore, to generalize the discussion, the procedure for the IP signaling protocol in the figures is the general connection setup procedure using confirmed service. In the figures, N-CONNECT and N-CONNECT-ACK show the resource reservation request and response messages, respectively; these messages are issued by the IP signaling entity. The SETUP, CALL PROC, CONN, and CONN ACK show the B-ISDN signaling messages.

**Overview of bottom-up approach.** An example procedure of SVC support for the bottom-up approach is shown in Fig. 2. After an SVC is established, the IP signaling entity on the called side watches for the arrival of the N-CONNECT message corresponding to the SVC. When it receives an N-CONNECT message, it confirms the existence of the SVC corresponding to it. Therefore, the SETUP message in the B-ISDN signaling must contain a session identifier and notify the



**Fig. 2.** Example procedure of SVC support for bottom-up approach

IP signaling entity of the arrival. If an N-CONNECT message does not arrive with a specified time from the SVC establishment, the IP signaling entity on the called side disconnects the SVC.

While this procedure is simple straight-forward scheme, it has some problems.

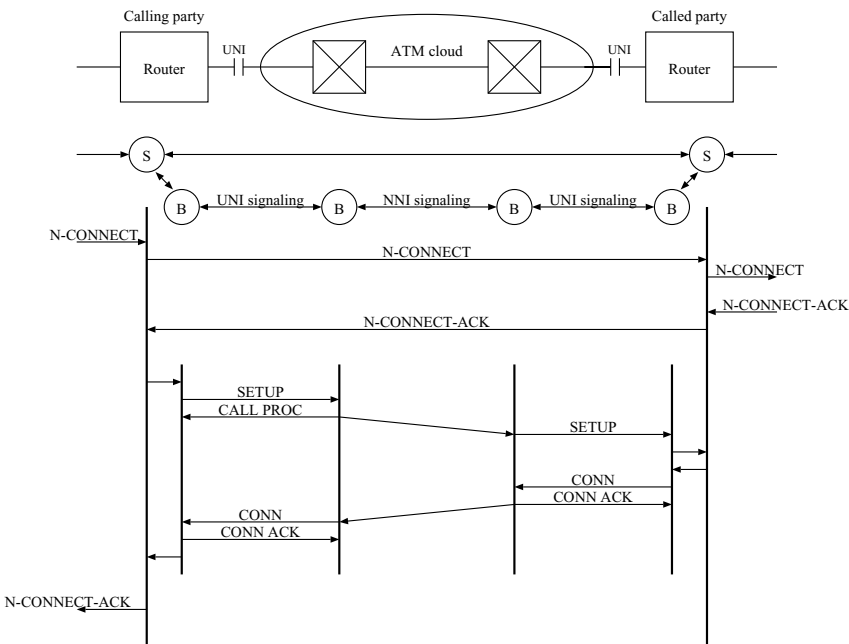
- If the IP signaling protocol supports negotiation, the SVC cannot reflect the negotiated parameters because the SVC is established before resource is reserved in the IP layer.
- If an N-CONNECT message is received from a user who is not allowed to access the IP layer, communication via the SVC may be possible until access denial is received from admission control. This is because after SVC establishment, admission control in the IP layer is initiated by the IP signaling protocol. There is thus a security problem.
- In the IP signaling entity on the called side, watchdog timer processing, which supervises the arrival of an N-CONNECT message, is needed after an SVC is established.

Note that in this example, the requester of the resource reservation and the calling party of the B-ISDN signaling are on the same side. However, in practical networks, it is possible that they are on opposite sides for administrative reasons.

If so, the B-ISDN signaling uses a callback procedure from the response side of the resource reservation. This scheme has the same problems as the one above.

The difference between PVC and SVC support for the bottom-up approach is that in PVC support, when an N-CONNECT message arrives at the IP signaling entity, it selects an available PVC, and does not establish an SVC. Therefore, it has the same problems as SVC support, except for the watchdog timer processing problem.

**Overview of top-down approach.** An example procedure of SVC support for the top-down approach is shown in Fig. 3. After resource is reserved in the



**Fig. 3.** Example procedure of SVC support for top-down approach

IP layer, the IP signaling entity on the called side watches for the arrival of a STEUP message corresponding to the reserved resource. When it receives a SETUP message, it confirms the existence of the reserved resource corresponding to the message. Therefore, the SETUP message in the B-ISDN signaling must contain a session identifier and notify the IP signaling entity of the arrival. If a SETUP message does not arrive with a specified time from the resource reservation in the IP layer, the IP signaling entity cancels the reservation.

This procedure has the following problems.

- If other network-layer protocols use ATM, it is possible that admission control for the ATM layer will fail due to insufficient resources, in spite of the resource reservation in the IP layer. This is because admission control for the ATM layer is initiated after the resource reservation.
- In the IP signaling entity on the called side, watchdog timer processing, which supervises the arrival of a SETUP message, is needed after resource reservation.

As in the previous example, the requester of the resource reservation and the calling party of the B-ISDN signaling are on the same side, but they may be on opposite sides. If so, the B-ISDN signaling uses a callback procedure from the response side of the resource reservation. This scheme has the same problems as the one above.

The difference between PVC and SVC support for the top-down approach is that in PVC support, when an N-CONNECT-ACK message arrives at the IP signaling entity, it selects an available PVC and does not establish an SVC. Therefore, it does not need watchdog timer processing. However, it encounters a similar problem with admission control: the IP signaling entity cannot select an available PVC due to exhaustion in spite of the resource reservation in the IP layer.

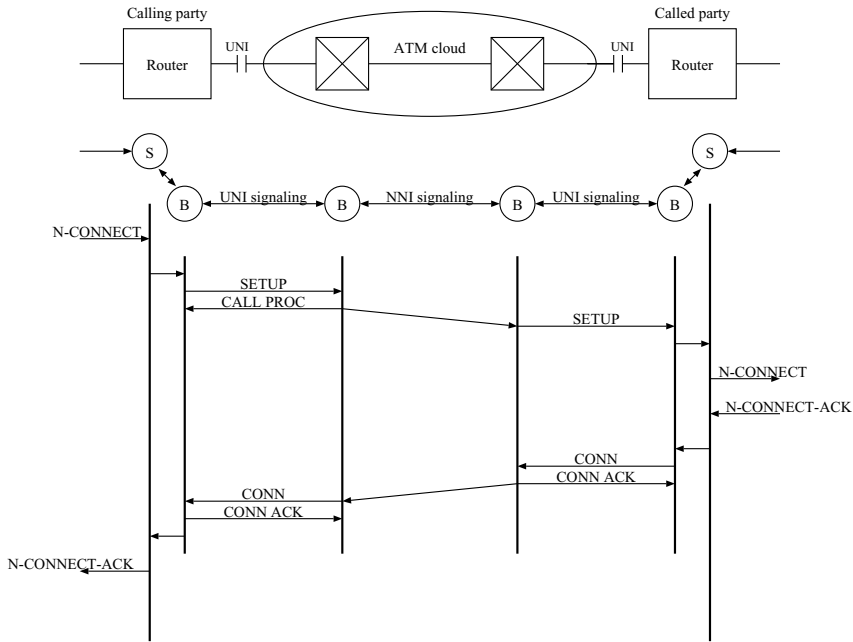
Note that in both the bottom-up and top-down approaches, if the relation between PVCs connecting routers and sessions changes dynamically, each PVC must have an identifier that is unique between routers, and the router that determines the relationship must notify the other router of the identifier. The IP signaling protocol must therefore be capable of transferring a PVC identifier.

**Overview of simultaneous approach.** An example procedure of SVC support for the simultaneous approach is shown in Fig. 4. In this approach, the IP signaling protocol is forwarded as an information element of the B-ISDN signaling, and the B-ISDN signaling entity on the called side notifies the IP signaling entity of the message. Therefore, with this approach, the B-ISDN signaling must be capable of transferring the IP signaling protocol. In addition, the SETUP message in the B-ISDN signaling must contain the session identifier, which is usually contained in the IP signaling protocol. This procedure is not applicable if the ATM network supports only PVC.

**Comparison of approaches.** While the bottom-up approach seems a natural one because it establishes the ATM connection first, it cannot support the negotiation function of the IP signaling protocol and it may have a security problem.

The top-down approach seems an unnatural one because it reserves resource in the IP layer first, but it does not have the problems of the bottom-up approach. However, it needs watchdog timer processing to supervise the arrival of a SETUP message, so its implementation is more complex.





**Fig. 4.** Example procedure of SVC support for simultaneous approach

The simultaneous approach does not have these problems and is easier to implement than the top-down approach. However, it is not applicable if the ATM network supports only PVC.

The simultaneous approach has many features comparable to those of the top-down approach, so it is the most appropriate if the ATM network supports SVC. Therefore, QoS-session signaling should be implemented using the simultaneous or top-down approach.

### 3.3 Requirements for B-ISDN Signaling

Similarly to the long-lived session signaling, if ATM SVC support for QoS-sensitive session is implemented using the bottom-up or top-down approach, the SETUP message in the B-ISDN signaling must be capable of transferring the session identifier. If the simultaneous approach is used, the B-ISDN signaling must be capable of transferring the IP signaling protocol.

### 3.4 Problems with Current B-ISDN Signaling and Solution

The Q.2931 and UNI 3.1 signaling protocol specifications do not contain an information element that could be used for transferring the IP signaling protocol.

The SCSI<sup>3</sup> and SIG 4.0 protocol specifications contain a UUS information element, which could be used for transferring the IP signaling protocol [10,20]. The UUS element enables the transfer of information between end-to-end users in an ATM network; it is an optional information element in the UNI signaling protocol. In SCSI and SIG 4.0, signaling messages transferred between end-to-end users may contain a UUS element, and the ATM network transfers this element transparently.

These specifications do not assign a protocol discriminator for the Internet protocol. Therefore, I proposed the assignment of the information field and protocol identifier in the Q.2957 UUS for the Internet protocol. In the new UUS recommendation, a protocol discriminator for the Internet protocol is assigned. (The problems and solution for the session identifier were described in section 2.)

## 4 Discussion

Depending on the IP signaling protocol architecture, if the IP signaling protocol supports uni-directional multicast communications, the simultaneous approach using the Q.2957 UUS element may have a problem.

A multicast tree consists of a sender, receivers, and routers corresponding to the root, leaves, and nodes, respectively. An ATM SVC connecting a sender and a router or two routers corresponds to a session, and this session may support several receivers. Because the UUS element transfers user-to-user information by using a B-ISDN signaling message, once an SVC is established, it cannot support transferring information until just before it is disconnected. Therefore, once an SVC connecting a sender and a router or two routers is established, this branch cannot support transferring the IP signaling protocol for additional receivers.

This problem does not occur if the IP signaling protocol architecture is as follows: when adding a new receiver to a session, if a router between the sender and receiver does not need to reserve a new resource, like RSVP, the IP signaling protocol is terminated at the router. This is because a branch between the sender and a router or two routers whose resource is already reserved, i.e., an SVC has been established, need not transfer the IP signaling protocol for the new receiver.

However, this problem occurs if the architecture is as follows: when adding a new receiver to a session, like ST2+, the IP signaling protocol must be exchanged between the sender and the receiver. Needless to say, this problem does not occur in a branch that supports only one receiver, i.e., an ATM SVC connecting the sender and a receiver or between a router and a receiver.

The UUS service 3 category may solve this problem. The Q.2957 UUS recommendation describes three service categories: service 1 transfers user-to-user information by using B-ISDN signaling messages, service 2 does it during the call-establishment phase, and service 3 does it while the call is active. However,

<sup>3</sup> Signalling Capability Set No. 1: covers ITU-T Recommendations Q.2931, Q.2951.X, Q.2955.1, and Q.2957, which are the basis of B-ISDN signaling.

Q.2957 provides detailed specifications for service 1 only, it does not specify services 2 and 3. Once the UUS protocol for service 3 is specified, this problem is solved because the IP signaling protocol can be transferred after call establishment.

The APM (application transport mechanism) may also solve this problem. The APM enables message exchange between any signaling entities; Q.2765 [23] describes an APM protocol between the NNIs in an B-ISDN. If an APM protocol between the UNIs of a B-ISDN is specified, this problem is solved because the IP signaling protocol can be transferred after call establishment.

## 5 Conclusion

In this paper I explained how the specifications, I proposed in the ITU-T and IETF, enable the B-ISDN signaling support to be used for long-lived and QoS-sensitive sessions.

Long-lived session requires that the B-ISDN signaling be capable of transferring a session identifier, which is currently supported only by proprietary protocols. By using the GIT information element, B-ISDN signaling can be used to support it.

To implement ATM VC support for QoS-sensitive session, I described three approaches: resource in the IP layer is reserved after the VC corresponding to the resource is established, the VC is established after the resource is reserved, or the resource is reserved and the VC is set up simultaneously. The first two approaches are applicable to SVC/PVC ATM networks, while SVC networks only support the third approach, whose implementation is the simplest.

In QoS-sensitive session using the first two and final approaches, the B-ISDN signaling must transfer the session identifier and the IP signaling protocol, respectively. Therefore, B-ISDN signaling based on current recommendations and standards cannot support QoS-sensitive session. These problems can be solved by using the GIT or UUS information element, as I have proposed. However, depending on the IP signaling protocol architecture, if the IP signaling protocol supports multicast communications, the final approach may not transfer the IP signaling protocol for additional receivers, and solutions using the UUS service 3 category or APM were discussed.

## Acknowledgments

I would like to thank Kenichi Kitami of the NTT Information Sharing Lab. Group, who is also the chair of ITU-T SG11 WP1, Shinichi Kuribayashi of the NTT Information Sharing Platform Labs., and Takumi Ohba of the NTT Network Service Systems Labs. for their valuable comments and discussions.

Also this specification is based on various discussions during the ST2+ over ATM project at the NTT Multimedia Joint Project with NACSIS. I would like to thank Professor Shoichiro Asano of the National Center for Science Information Systems for his invaluable advice in this area.

## References

1. L. Delgrossi and L. Berger, Ed., "Internet Stream Protocol Version 2 (ST2) Protocol Specification - Version ST2+," RFC 1819, August 1995.
2. R. Braden Ed., "Resource ReSerVation Protocol (RSVP)-Version 1 Functional Specification," RFC 2205, September 1997.
3. P. Newman, T. Lyon, and G. Minshall, "Flow Labelled IP: A Connectionless Approach to ATM," Proc. IEEE Infocom, March 1996.
4. Y. Katsube, et al., "Toshiba's Router Architecture Extensions for ATM : Overview," RFC 2098, February 1997.
5. M. Suzuki, "ST2+ over ATM Protocol Specification - UNI 3.1 Version," RFC 2383, August 1998.
6. E. Crawley Ed., "A Framework for Integrated Services and RSVP over ATM," RFC 2382, August 1998.
7. M. Laubach and J. Halpern, "Classical IP and ARP over ATM," RFC 2225, April 1998.
8. The ATM Forum, "LAN Emulation Over ATM Version 2 - LUNI Specification," July 1997.
9. ITU-T, "Generic Identifier Transport," Draft ITU-T New Recommendation Q.2941.1, September 1997.
10. ITU-T, "User-to-User Signalling (UUS)," ITU-T Recommendation Q.2957, February 1995.
11. ITU-T, "Generic Identifier Transport Extensions," Draft ITU-T New Recommendation Q.2941.2, November 1999.
12. ITU-T, "User-to-User Signalling (UUS)," Draft ITU-T Recommendation Q.2957 Amendment 1, November 1999.
13. M. Suzuki, "The Assignment of the Information Field and Protocol Identifier in the Q.2941 Generic Identifier and Q.2957 User-to-user Signaling for the Internet Protocol," Internet Draft, January 2000.
14. J. Postel Ed., "Internet Protocol," RFC 791, September 1981.
15. S. Deering and R. Hinden, "Internet Protocol, Version 6 (IPv6) Specification," RFC 2460, December 1998.
16. J. Postel, "User Datagram Protocol," RFC 768, August 1980.
17. J. Postel Ed., "Transmission Control Protocol," RFC 793, September 1981.
18. ITU-T, "User-Network Interface (UNI) Layer 3 Specification for Basic Call/Connection Control," ITU-T Recommendation Q.2931, September 1995.
19. The ATM Forum, "ATM User-Network Interface Specification Version 3.1," September 1994.
20. The ATM Forum, "ATM User-Network Interface (UNI) Signaling Specification Version 4.0," July 1996.
21. S. Damaskos and A. Gavras, "Connection Oriented Protocols over ATM: A case study," Proc. SPIE, Vol. 2188, pp.226-278, February 1994.
22. ITU-T, "Specification of a Synchronization and Coordination Function for the Provision of the OSI Connection-mode Network Service in an ISDN Environment," ITU-T Recommendation Q.923, February 1995.
23. ITU-T, "Signaling System No.7 B-ISDN User Part, Application Transport Mechanism," Draft ITU-T Recommendation Q.2765, July 1999.

# Mapping an Internet Assured Service on the GFR ATM Service

Fernando Cerdán and Olga Casals

Polytechnic University of Catalonia, Computer Department Architecture,  
C/ Jordi Girona 1-3, E-08071 Barcelona Spain  
{fernando,olga}@ac.upc.es

**Abstract.** As Internet is rapidly growing and receiving traffic from multimedia applications which are sensitive to available bandwidth and delay experienced in the network, there is a strong need for quality of service (QoS) support. The Integrated and Differentiated Service models are two approaches for adding QoS to Internet. The Assured Service is an end-to-end service based on the Differentiated Service architecture. Since ATM is widely deployed in Internet backbones, traffic management for Internet traffic over ATM is becoming an important issue. In this paper we present a simulation analysis of an Assured service mapped on the ATM GFR service category. We study the problem of guarantying individual target rates to an aggregated IP stream. Aggregated streams are mapped with a MCR according to the QoS requested.

## 1 Introduction

With the increasing interest in multimedia and the tremendous growth of word wide web (WWW), Internet is receiving a large amount of this type of traffic. Multimedia applications are sensitive to available bandwidth and delay experienced in the network. Internet must be developed in order that customers are able to obtain different quality of service according to their needs and willingness to pay. At this moment there are two approaches known as Integrated services [1] and Differentiated Services [2] respectively.

The Differentiated Services approach defines a group of mechanisms to treat packets with different priorities according to the information carried in the Diff. Serv. field of the IP packet header. Packets are classified and marked to receive a particular per-hop forwarding behaviour (PHB) on nodes along their path. Complex classification and conditioning functions (metering, marking, shaping) need only to be implemented at boundary nodes. Interior nodes perform a set of forwarding behaviours (PHBs) to aggregates of traffic which have been appropriately marked. A set of PHBs are being standardised at the IETF to develop end-to-end differentiated services. Two PHBs are currently in wide discussion: the Expedited Forwarding PHB (EF-PHB) and the Assured Forwarding PHB (AF-PHB). The Assured Forwarding Per Hop Behaviour (AF-PHB) [6] currently allows a domain server (DS) provider to offer for general use, until four different levels of assurance (AF classes), to deliver IP packets. IP packets must belong to one of the AF classes. Within each AF class, either

the user or the DS domain can mark an IP packet with three levels of loss precedence. The AF-PHB is used to build the end-to-end Assured Service (AS).

We shall consider a simplified form of the assured service which is similar to the original model proposed by Clark [4,9]. The user of an AS expects a low dropping probability as long as it sends traffic within the expected capacity profile. Packets of individual flows are marked as IN (in profile) or OUT (exceeding the profile). Routers inside the network do not distinguish packets of individual flows but they implement a preferential drop mechanism giving preference to IN packets. With appropriate network provisioning it is expected that this could result in bandwidth guarantees.

Since ATM is widely deployed in Internet backbones, traffic management for Internet traffic over ATM is becoming an important issue. ATM cells flow along virtual channels (VCs) previously set up according to admission control parameters. Nevertheless TCP traffic with a single TCP per VC will not be used in Internet backbones. Many TCP connections will be multiplexed on a single VC and the control of individual TCP traffic characteristics will not be possible. Aggregation of Internet (IP) flows is necessary for scalability as well as for overhead and complexity reduction. The Differentiated Service model commented above seems to be the best way of providing QoS to aggregated flows.

We will assume one AF class (Assured Service) based on the TSW-RIO framework described in [4] with only two loss precedence levels, IN and OUT packets. In this case and taking into account that ATM has two drop preferences through the cell loss priority bit, the mapping of differentiated services to ATM may be feasible.

The assured service class (AS) is intended to give the customer the assurance of a minimum average throughput, the target rate, even during periods of congestion. The remaining bandwidth should be shared in a fair manner among all competing sources. In ATM the GFR service category provides the user with a minimum service rate guarantee under the assumption of a given maximum frame size. The service allows the user to send traffic in excess of its guaranteed service rate (MCR), but these frames will only be delivered within the limits of available bandwidth.

In this paper we deal with the problem of mapping an Assured Service on the GFR service category. We consider the situation where the sources of two IP networks reach their destinations through an ATM network. Individual flows contract a profile given by target rates which are monitored and aggregated in a RIO router. Aggregated flows are mapped on the ATM network through the GFR service category with a MCR. The MCR is the sum of individual contracts. Implementation of the GFR service is based on either the DFBA mechanism [12], or the GFS scheduling first proposed in [7]. The rest of the paper is organized as follows: In section 2 we give a description of an assured service architecture. We present the configuration that will be used along the paper. Section 3 is dedicated to provide an overview of GFR scheduling mechanisms. We compare the performance of DFBA and GFS. In section 4 we present the simulation environment with two IP networks connected to an ATM network. The first IP network uses an Assured Service architecture while the second is best effort. Simulation results with DFBA and GFS scheduling in the ATM switch are analysed. Discrimination of IN and OUT packets in the aggregated stream is a key issue for providing individual target rates. Finally in section 5 we give some conclusions.

## 2 Description of the Assured Service Architecture

In the current assured service architecture there are two main components (see Figure 1). The first is the profiler used to distinguish assured packets (IN) from excess packets (OUT). The token bucket and the TSW are the two most commonly used profilers. The second component is the router. The queue management most commonly implemented in the routers has twin RED algorithms to preferentially drop OUT packets. The mechanism is known as RIO (RED with IN and OUT) [4].

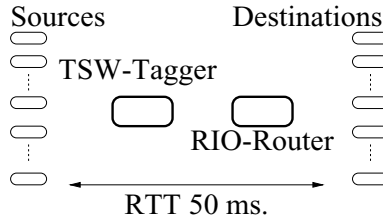


Fig. 1. Simulated assured service architecture

The TSW profiler estimates the average rate upon each packet arrival and decays the past history over a  $W\_length$  period of time.  $W\_length$  is a constant parameter pre-configured initially. The algorithm for rate estimation in the TSW profiler is fully described in [4]. Ideally the profiler should keep a TCP connection oscillating between 0.66 Tr (Target rate) and 1.33 Tr, so that on average the connection can achieve the target rate. Packets are marked as OUT with probability  $p$  (equation (1)), when the average rate exceeds a certain threshold.

$$p \equiv \frac{\text{Average\_rate} - \text{Target\_rate}}{\text{Average\_rate}} \quad (1)$$

A TSW profiler interacts with the RIO router, which is dropping packets according to its congestion state. Given the amount of parameters involved and the bursty nature of the TCP traffic one of the main difficulties in designing RIO-TSW is to choose the right values of these parameters.

In terms of tagging the TSW algorithm can follow two approaches. The first remembers a long past history and packets are marked as OUT with probability  $p$  when the average rate exceeds the target rate. In the second approach TSW remembers a short period of time, on the order of a RTT, and packets are marked as OUT with probability  $p$  when the average rate exceeds a threshold of ( $\beta \cdot \text{target rate}$ ) with  $\beta > 1$ . In general TSW is quite sensitive to this threshold due to the TCP congestion control mechanism which determines the instantaneous TCP sending rate. The instantaneous TCP sending rate may depend on many parameters such as the RTT, the retransmission timer granularity, MSS, and the TCP congestion control when the router drops packets.

To show that, let's consider the examples of Table 1, case A and case B, where 8 and 10 respectively TCP sources with different target rates are multiplexed in a RIO

router using the configuration of Figure 1. This configuration is based on the fact that ATM is the physical layer. IP over ATM uses a maximum segment size of 9180 bytes which has been approximated by 200 cells. In case A the sum of the target rates is 60% of the link rate, while in case B it is 90%. The RIO parameters (Min, Max, Pmax) are 40/70/0.02 for IN packets and 10/40/0.2 for OUT packets.

In both cases we observe that with a good tuning of all parameters the average number of IN packets and the average throughput obtained match reasonably well the target rate and the fair throughput of the connection respectively. The fair throughput is defined in equation (2).

$$\text{Fair\_Th}(i) \equiv \text{Target\_rate}(i) + \frac{\text{Link\_rate} - \sum_{i=1}^{\text{N}^\circ \text{Connections}} \text{Target\_rate}(i)}{\text{N}^\circ \text{Connections}} \quad (2)$$

The average number of packets marked as IN by the TSW profiler and the obtained average throughput can be obtained with a high degree of approximation using either a value of  $\beta = 1.33$  and  $W\_length = 200\text{ms}$  or a value of  $\beta = 2.5$  and  $W\_length = 50\text{ ms}$ .

**Table 1.** TSW-RIO Performance with 8 sources (Case A) and 10 sources (Case B) with different target rates. All measured values in Mbps.

TCP RENO sources with MSS = 9180 bytes, LINK = 33 Mbps									
Flow number		Target rate		Average IN Throughput		Average Throughput		Fair Throughput	
A	B	A	B	A	B	A	B	A	B
0	0	1	1	0.87	0.87	2.69	1.50	2.625	1.30
1	1	1	1	0.92	0.87	2.37	1.44	2.625	1.30
2	2	2	2	1.94	1.65	3.21	2.10	3.625	2.30
3	3	2	2	1.87	1.71	2.98	2.21	3.625	2.30
4	4	3	3	3.15	2.51	4.38	2.85	4.625	3.30
5	5	3	3	3.36	2.52	4.65	2.90	4.625	3.30
6	6	4	4	4.47	3.98	5.54	4.33	5.625	4.30
7	7	4	4	4.61	3.96	5.52	4.32	5.625	4.30
	8		5		5.09		5.40		5.30
	9		5		5.13		5.43		5.30

The above examples are two cases where a good combination of parameters makes the configuration work as expected. Nevertheless this configuration may fail when either the RTTs are different [4] or it is applied to the whole Internet [10]. The selection of an optimum set of parameters is a very difficult task which is out of the scope of this paper. Given a configuration where the average number of IN packets in the network is quite close to the target rates, our interest focus on how these IN and OUT packets are handled through the ATM network and how this treatment affects the end-to-end performance of individual connections, i.e. the target rates. The above configurations will be used in the next sections.



### 3 Summary of Scheduling Mechanisms for GFR

The GFR traffic contract uses four parameters. The first two are the PCR (Peak Cell Rate) and a CDVT (Cell Delay Variation Tolerance) for conformance purpose at the UNI (User Network Interface). This conformance is carried out by the ATM GCRA(1/PCR,CDVT). The third is the MCR (Minimum Cell Rate) which corresponds to the amount of bandwidth that needs to be guaranteed. Finally the forth is the Maximum Frame Size accepted by both ends of the link.

Another important issue is the identification of the frames to which the service guarantees apply. Frames which are not identified for service guarantee may be tagged with low priority and they should be served or discarded according to network congestion. The proposed implementations of GFR at a switch have been studied with TCP connections, since today a large number of applications rely on TCP/IP. In the following we describe four examples of GFR mechanisms (FIFO, WFQ, DFBA and GFS). The first three are included in the ATM Forum.

**FIFO :** The FIFO-based implementation assumes that the modified GCRA will be used to identify the frames to which the service guarantees apply. Tagging of excess traffic can be performed either at the network access point or at the switch element itself. This implementation is quite simple, but this simplicity comes at some cost. In particular, fairness in the usage of excess bandwidth can not be achieved. The study with TCP sources performed in [11] shows that the performance of TCP is never satisfactory even in the case of a LAN. When the switch buffers are infinite, the bandwidth reservations, (MCR), do not have any influence on the throughput achieved by TCP, since the link is equally shared among all connections. When the switch buffers are finite and losses occur, a TCP source is not able to get its reserved bandwidth.

**WFQ :** The WFQ implementation relays on per-VC accounting and per-VC scheduling. A WFQ scheduler is able to enforce a minimum cell rate for each VC. If the network does not perform tagging of VC's, but it uses the tagging information in the cells for traffic management, then per-VC accounting of untagged cells becomes necessary for providing minimum guarantees to the untagged stream.

Unlike the FIFO based implementation the study done in [11] and [8] with TCP sources using WFQ scheduling shows that the performance in terms of throughput is satisfactory when no losses occur. However, this implementation allocates bandwidth according to the weights, which depend on the GFR contracts (MCR). While the GFR service category is not expected to provide a fair treatment in terms of transmission delay, this allocation can delay in excess frames of sources with very low GFR contracts. Other simulation results in WAN and LAN environments with limited buffers, i.e. when losses occur reveal that TCP performance is not satisfactory in using its reserved bandwidth.

**DFBA :** DFBA is a buffer management scheme that provides minimum rate guarantees to TCP/IP traffic. This scheme is based on the principle that TCP rates can be controlled by adjusting the window size of the TCP connection. This is achieved by discarding segments at specific congestion window values. DFBA uses per-VC accounting as well as static and dynamic thresholds in a FIFO buffer, which estimate the bandwidth used by the connection. Thereby, if the buffer occupancy of each active VC is maintained at a desired threshold, then the output rate of each VC can also be



Table 2 summarises a comparative study between DFBA and GFS. We have multiplexed 5 VC's carrying each one 3 TCP sources in an ATM switch. The ATM switch implements either the DFBA or GFS algorithm assuming that all frames are CLP 0. DFBA main parameters have been chosen based on suggestions given in [12]. The link bandwidth is 155.52 Mbps and the MCR allocations for each VC are 5, 10, 15, 20 and 25 Mbps respectively. The Maximum buffer size is 25000 cells. DFBA thresholds are  $L = 12500$  cells and  $H = 22500$  cells, while in GFS the HBO threshold is 20000 cells. TCP sources are greedy and they are based on the RENO implementation [13], with a MSS of 1024 bytes.

As it is shown in ,Table 2 GFS which was originally proposed to work with per-VC queuing can also provide MCR guarantees when several connections are aggregated to a single VC. Furthermore DFBA and GFS divide the excess bandwidth among the different connections in the same way, as indicated in the last column of Table 2.

4 Performance of the Assured Service Through an ATM Network

Given the similar specifications of the Assured Service and the GFR ATM service category, a set of simulations have been carried out in order to study the feasibility of mapping the Assured service on the GFR service. Simulations are based on the network configuration presented in Figure 2.

Users in the first network (IP network 1 in figure 2) follow the assured service model described in section 2 with two precedence levels based on the TSW-RIO configuration. Each user in this network has contracted a different target rate which is provided by its DS provider. Users in the second network (IP network 2 in figure 2) follow the current best effort network model of the internet without any guaranties of QoS. Each flow gets the maximum bandwidth available based on the congestion state of the path. The router in the network implements the RED algorithm [5].

Aggregated traffic streams are multiplexed in the ATM network through the GFR service. The ATM switch implements either the GFS or DFBA scheduling mechanisms. Each aggregated stream is assigned a VC with a MCR contract.

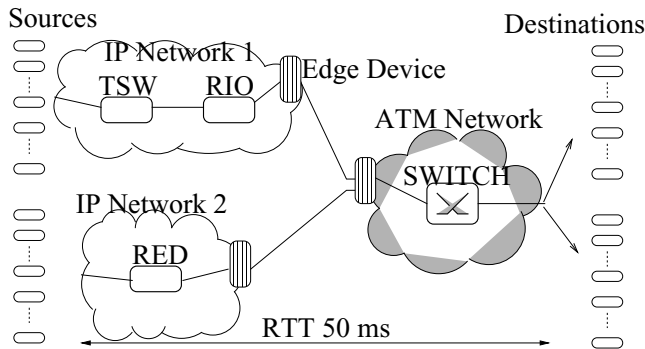


Fig. 2. Simulated assured service architecture through the GFR ATM service

The aggregated traffic stream from IP network 2 has two options when entering the ATM network. Either it contracts a MCR null, so that it will share the leftover bandwidth with the flow from IP network 1, or it contracts a certain MCR. In the former case individual flows will get a proportional part of the leftover bandwidth. In the later individual flows should get a proportional part of the contracted plus the leftover bandwidth. The aggregated traffic stream from IP network 2 has only OUT packets. This could be a problem for the scheduling mechanism to guarantee the MCR of the flow since OUT packets will be treated as excess bandwidth. This issue will be analysed in the following sections.

The configuration of Figure 2 assumes IP over ATM with a link bandwidth of 33 Mbps in all paths. IP network 1 has eight Assured Service sources with different target rates from 1Mbps to 4 Mbps. They are multiplexed in a RIO router with the same parameters as described in the configuration of Figure 1. So we expect the same average number of IN and OUT packets as shown in Table 1 at the edge of IP network 1. This assumption is valid as long as the influence of congestion in the ATM network is not meaningful. IP network 2 has also 8 sources multiplexed in a RED router with parameters 10/40/0.02. As commented above, packets from this network are considered as OUT packets when entering the ATM network.

The aggregated flow from IP network 1 is assigned a MCR equal to the sum of all its target rates, 20 Mbps. We consider that the aggregated flow from IP network 2 contracts a MCR of 8Mbps in order to provide 1Mbps to each user.

#### 4.1 Mapping on GFR Without IN and OUT Packet Discrimination

All proposed implementations of the GFR service included in the ATM forum (FIFO, WFQ, DFBA) and GFS have a common characteristic. The buffer is shared by all flows, and it contains common thresholds. Isolation among the different flows is carried out by means of additional mechanisms such as policing or per-VC accounting.

When using WFQ and GFS without tagging the excess traffic only one threshold is needed. Given that the thresholds are common for all flows, isolation among them is not as good as it would be desirable. WFQ presents serious problems related to the ones which have been reported in [8]. Only GFS and DFBA can provide MCRs to all flows. Nevertheless in the current context the number of IN packets in the flow provides the target rate. If the buffer is congested, and one of the flows is sending more OUT packets than the others do, packets will be discarded from all flows without taking into account if they are IN or OUT packets. It seems clear that under these conditions it will be difficult to get individual target rates of each flow through the ATM network.

Simulation results presented in Table 3 show that the obtained average throughputs of individual flows in IP network 1 are quite different from the expected fair throughput. Actually, these flows get a throughput which is a proportional share of the MCR contracted plus the leftover bandwidth.

We may consider an infinite ATM buffer in order to ensure that all IN packets will be delivered. Simulation results of Table 4 show that in this case the buffer utilisation is too high (we measured an output load close to 0.95), so the queuing delays

increased significantly causing some TCP sources to time out. In these cases the rest of the sources take advantage to increase throughput above their fair quota.

**Table 3.** Finite ATM buffer (GFS : QMAX 2000 cells, HBO 1000cells ; DFBA : H 1800 cells, L 1000 cells). Measured values in Mbps

TCP RENO sources with MSS = 9180 bytes, LINK = 33 Mbps									
Flow number		Target rate		Average Throughput GFS		Fair Throughput		Average Throughput DFBA	
IP1	IP2	IP1	IP2	IP1	IP2	IP1	IP2	IP1	IP2
0	8	1	-	2.61	1.40	1.30	1.30	2.50	1.50
1	9	1	-	2.66	1.37	1.30	1.30	2.63	1.38
2	10	2	-	2.81	1.29	2.30	1.30	2.76	1.50
3	11	2	-	2.76	1.31	2.30	1.30	2.46	1.49
4	12	3	-	2.66	1.31	3.30	1.30	2.41	1.60
5	13	3	-	2.73	1.28	3.30	1.30	2.71	1.56
6	14	4	-	2.59	1.36	4.30	1.30	2.53	1.57
7	15	4	-	2.59	1.13	4.30	1.30	2.65	1.18

4.2 Mapping on GFR with IN and OUT Packet Discrimination. GFS Case

In this section we try to solve the problems described above, in order to guarantee the target rate of each individual connection. Discrimination between IN and OUT packets is a key issue when buffering incoming packets. IN and OUT packets are mapped in the GFR service category through the CLP bit as CLP 0 and CLP 1 respectively. The pseudocode for cell acceptance in GFS scheduling accepts a packet as eligible for services guarantees as long as the counter `TOKEN_CNTR`  $\geq$  `Maximum_packet_size`. In all other cases the packet is considered as excess traffic. If we do not impose that eligible packets must be also IN packets the scheduler may choose OUT packets as eligible. In case of congestion IN packets of the aggregated stream may be dropped affecting the target rate of the individual connections. The GFS scheduling must be modified in this way to preserve all individual target rates in an aggregated stream (see line 3 in Figure 3).

In this case, as shown in Table 5, with the modified GFS algorithm all individual connections of IP network 1 can get their target rates. Connections with the lowest target rates benefit more from the excess bandwidth. Nevertheless sources of IP network 2 manage to get their contracted MCR. The main reason of this behaviour is that all packets from IP network 2 are OUT packets and they are considered as excess bandwidth, i.e the scheduler does not take into account the MCR contracted. Thereby the average throughput presented in Table 5 for each flow in IP network 2 is correct. The current excess bandwidth is  $33-20=13$  Mbps, instead of the  $33-28=5$  Mbps considered. See the column labelled “*Current fair throughput*” in Table 5.

**Table 4.** Infinite ATM Buffer. Measured values in Mbps

TCP RENO sources with MSS = 9180 bytes, LINK = 33 Mbps									
Flow number		Target rate		Average Throughput GFS		Fair Throughput		Average Throughput DFBA	
IP1	IP2	IP1	IP2	IP1	IP2	IP1	IP2	IP1	IP2
0	8	1	-	3.31	1.75	1.30	1.30	2.52	1.27
1	9	1	-	3.17	0.55	1.30	1.30	2.51	2.29
2	10	2	-	3.34	0.70	2.30	1.30	1.26	2.25
3	11	2	-	3.23	0.65	2.30	1.30	1.28	2.32
4	12	3	-	1.65	1.69	3.30	1.30	2.51	2.37
5	13	3	-	3.37	1.70	3.30	1.30	1.31	2.29
6	14	4	-	0.93	1.74	4.30	1.30	2.51	2.42
7	15	4	-	3.46	1.69	4.30	1.30	2.51	1.29

**Table 5.** Simulation results with modified GFS. Finite ATM buffer : QMAX 2000 cells, HBO 1000 cells. Measured values in Mbps.

TCP RENO sources with MSS = 9180 bytes, LINK = 33 Mbps									
Flow number		Target rate		Avg. Thput. GFS		Fair Throughput		Current Fair Thput. GFS	
IP1	IP2	IP1	IP2	IP1	IP2	IP1	IP2	IP1	IP2
0	8	1	-	2.07	0.91	1.30	1.30	1.81	0.81
1	9	1	-	2.12	0.86	1.30	1.30	1.81	0.81
2	10	2	-	2.90	0.87	2.30	1.30	2.81	0.81
3	11	2	-	2.92	0.73	2.30	1.30	2.81	0.81
4	12	3	-	3.35	0.83	3.30	1.30	3.81	0.81
5	13	3	-	3.65	0.91	3.30	1.30	3.81	0.81
6	14	4	-	4.50	0.90	4.30	1.30	4.81	0.81
7	15	4	-	4.43	0.67	4.30	1.30	4.81	0.81

### 4.3 Mapping on GFR Service with IN and OUT Packet Discrimination. DFBA Case

Now we analyse the above case but using the DFBA scheduling algorithm. As in the GFS case, IN and OUT packets are mapped with CLP 0 and CLP 1 respectively. From the simulation results presented in Table 6, we observe that all flows of IP network 1 can get their target rate although connections with the lowest target rates benefit more from the excess bandwidth as in the GFS case. The individual flows of IP network 2 do not get the expected throughput. The MCR contracted is reached but they can not benefit from the excess bandwidth. Since all packets are OUT (CLP 1) and given the high link utilisation they are always dropped when the buffer size is above the L threshold. These drops provoke idle periods that reduce congestion in the buffer. In this situation OUT packets of IP network 1, which arrive more spaced, can use the excess bandwidth. This effect is considerably stronger if we reduce the L threshold value in the buffer (see last column in Table 6).

```

1   if (First_cell_of_frame)
2   {
3   if (TOKEN_CNTR(j) >=
      Maximum_frame_size(j)) && CLP 0
4   {
5       c = 2
6       if (QT<QMAX)
7       {
8           store_cell(VC_j)
9           add_reference(j,Global_FIFO)
10          TOKEN_CNTR(j)= TOKEN_CNTR(j) -1
11          QT=QT+1
12          EPD(j)= 1
13      }
14      else /*There is no buffer space */
15      {
16          TOKEN_CNTR(j)= TOKEN_CNTR(j) -1
17          EPD(j)= 0 /*Discard*/
18      }
19  }
20  else /* There is no enough tokens for flow j */
21  {
22      .....

```

**Fig. 3.** Fragment of the Pseudocode of the GFS scheduling algorithm modified to preferably store the IN packets of an aggregated stream.

**Table 6.** Simulation results with DFBA. Buffer dimensioning influences the obtained throughputs. In the first case, labelled DFBA: H = 3600 cells, L = 2000 cells. In the second case, labelled DFBA-2, H = 1800 cells, L = 1000 cells Measured values in Mbps.

TCP RENO sources with MSS = 9180 bytes, LINK = 33 Mbps									
Flow number		Target rate		Avg. Throug. DFBA		Fair Thougput		Avg. Throug. DFBA-2	
IP1	IP2	IP1	IP2	IP1	IP2	IP1	IP2	IP1	IP2
0	8	1	-	2.06	1.00	1.30	1.30	2.57	0.17
1	9	1	-	2.02	1.01	1.30	1.30	2.40	0.45
2	10	2	-	2.69	1.09	2.30	1.30	3.23	0.45
3	11	2	-	2.60	0.99	2.30	1.30	3.18	0.32
4	12	3	-	3.30	1.12	3.30	1.30	4.16	0.38
5	13	3	-	3.27	1.07	3.30	1.30	3.82	0.47
6	14	4	-	3.97	1.09	4.30	1.30	5.08	0.17
7	15	4	-	4.41	1.11	4.30	1.30	5.31	0.42

5 Conclusions

A simulation study of mapping an IP Assured Service on the GFR ATM service category has been done. An Assured Service architecture based on the TSW-RIO configuration with two levels of drop precedence (IN and OUT packets) has been used in order to provide an average number of IN packets in the network. This

number of IN packets should correspond to the target rate contracted by individual users. The TSW-RIO configuration involves a high number of parameters that may be difficult to tune. In our simulations we have considered a case where these parameters are well tuned. The aggregated assured service stream has been multiplexed in an ATM switch with an aggregated stream from another IP network based on the current best effort service. Assuming that all packets from the best effort network are considered as OUT packets, we have studied their influence in the target rate of the assured service connections. We have shown that the assured service connections can not get their target rates if the GFR scheduling algorithm (DFBA or GFS) do not distinguish between CLP 0 (IN) and CLP 1 (OUT) frames. Target rates can be achieved in DFBA as it is with CLP bit consideration. GFS requires a slight modification of the pseudocode for cell acceptance. Further studies are being carried out to have a deeper understanding of the influence of the parameters of conditioning functions and router schemes on performance characteristics.

## References

1. R. Braden and D. Clark, "Integrated Services in the Internet Architecture: an Overview" RFC 1633
2. S. Blake, D. Black, M. Carlson, E. Davies, Z. Wang and W. Weiss "An Architecture for Differentiated Services" RFC 2475
3. S. Fahmy, R. Jain, S. Rabie, R. Goyal, B. Vandalore "Quality of Service for Internet Traffic over ATM Service Categories" Computer Communications special issue on enterprise networks.
4. D. Clark and W. Fang "Explicit Allocation of Best-Effort Packet Delivery Service" IEEE/ACM Transactions on Networking, VOL 6 No. 4 August 1998
5. S. Floyd and V. Jacobson "Random Early Detection Gateways for congestion Avoidance" IEEE/ACM Transactions on Networking, August 1993
6. J. Heinanen, F. Baker, W. Weiss and J. Wroclawski "Assured Forwarding PHB Group" RFC 2597
7. F. Cerdán and O. Casals "A Per\_VC Global FIFO Scheduling Algorithm for Implementing the new ATM GFR Service" MMNS'98 Versailles FRANCE November 1998
8. F. Cerdán and O. Casals "Performance of Different TCP Implementations over the GFR Service Category" Seventh IFIP Workshop on Performance Modelling and Evaluation of ATM/IP Networks, Antwerp, BELGIUM June 28-30 1999
9. D. Clark, J. Wroclawski "An Approach to Service Allocation in the Internet" Internet Draft draft-clark-diff-svc-alloc-00.txt July 1997
10. J. Ibanez, K. Nichols "Preliminary Simulation Evaluation of an Assured Service" draft-ibanez-diffserv-assured-eval-00.txt, 1998
11. O. Bonaventure "A simulation study of TCP with the proposed GFR service category" High-Performance Networks for Multimedia Applications, June 1997, Schlob Dagstuhl, Germany
12. R. Goyal, R. Jain, S. Fahmy and B. Vandalore "Providing Rate Guarantees to TCP over the ATM GFR Service" Proceedings of 23rd Annual Conference on Local Computer Networks 1998(LCN'98), Lowell, MA, October 11-14, 1998, pp. 390-398,
13. W. Stevens "TCP Slow Start, Congestion Avoidance, Fast Retransmit and Fast Recovery algorithms" RFC 2581



# Real-Time Cell Arrival Sequence Estimation and Simulator for IP/ATM Networks \*

Hiroshi Saito<sup>1</sup> Toshiaki Tsuchiya<sup>1</sup>, Gyula Marosi<sup>2</sup>, Gyorgy Horvath<sup>2</sup>,  
Peter Tatai<sup>2</sup>, and Shoichiro Asano<sup>3</sup>

<sup>1</sup> NTT Service Integration Laboratories

<sup>2</sup> Technical University of Budapest

<sup>3</sup> NACSIS (National Center for Science and Information Systems)

**Abstract.** We have developed a new traffic measuring tool and applied it to the real-time simulation of a network. It monitors IP traffic on an ATM link and continuously transfers the length and timestamp of each IP packet to a post-processing system. The post-processing system receives the data, estimates the cell's arrival epoch at the transmission queue of the ATM link, and simulates the queueing behavior on-line if conditions differ from those of the actual system. The measuring tool and real-time simulation represent a new approach to traffic engineering. A new estimation problem, the arrival sequence estimation, is shown and some algorithms are proposed and evaluated.

## 1 Introduction

Internet traffic is growing rapidly world-wide, and the proliferation of new applications is causing its characteristics to change. Meanwhile, Asynchronous Transfer Mode (ATM) has begun to be deployed in internet backbone networks, several types of WAN services, and fast broadband private networks for companies within a group. In such new telecommunication environments, traffic engineering faces two difficulties. One is the limited capability to monitor and measure traffic in network elements such as ATM switches and routers. While capturing traffic characteristics is one of the most important issues for economical development and evaluation of new technologies, extra functions in network elements to monitor and measure traffic would raise the cost of the network element and reduce the processing power for call control and/or packet forwarding. So, existing network elements do not give traffic engineers all the information about the traffic characteristics that they would like, while the traffic characteristics are becoming more and more complicated and the link and processor capacities are increasing. Even when we can monitor traffic, traffic engineering (including traffic control, management, and dimensioning) is still a challenge in an ATM backbone for Internet traffic. This is the second difficulty. In particular, empirical tests (with theoretical background) are more preferable than pure theoretical

---

\* Hiroshi Saito, NTT Service Integration Labs., 3-9-11, Midori-cho, Musashino-shi, Tokyo, 180-8585, Japan. E-mail: [saito.hiroshi@lab.ntt.co.jp](mailto:saito.hiroshi@lab.ntt.co.jp), URL: <http://www.hslab.tnl.ntt.co.jp>

approaches in traffic engineering these days. Thus, modeling and characterizing traffic may not be enough to show the effectiveness of a proposed traffic engineering scheme. This paper shows how these two challenging issues can be overcome by developing traffic measuring tools outside network elements and passing its data to a real-time on-line simulator.

This kind of approach has received attention recently. DARPA (Defense Advanced Research Projects Agency), through its NGI (Next Generation Internet) project, aims at real-time network simulations as part of network engineering [11]. Hewlett Packard has also announced that NetMetrix, one of its products, can be combined with a simulation tool [12]. In practice, it seems to be different from our tool because we continuously monitor traffic and give every packet a timestamp, and the simulator uses the timestamp sequence continuously.

Overall, we propose traffic engineering steps using real-time on-line simulation: In the first step, the number of network resources such as link capacity or buffer size is roughly dimensioned by a certain method. In the second step, the dimensioned number of network resources is set in the simulator using the actual traffic data continuously. Instead of the dimensioned number, the number planned in the following year's budget, for example, can be used. In the third step, the simulator checks whether the dimensioned or planned number of network resources is appropriate. The fourth step yields the number of network resources that was dimensioned/planned, after it has been checked and modified if necessary. (Normally, the dimensioned or planned number may need a certain margin to accommodate the traffic growth by the time the updated network resources are provided. If we can do these four steps in a short period of time and repeat them, we can implement the self-sizing network concept, which can adapt its network size to the ever-changing traffic [5], [6], [7], [8], [9].)

## 2 Measuring Tool

We developed a new traffic measuring tool called CapTie (capturing traffic while being tied to a post-processing computer) on the real-time OS called VxWorks [10]. (The first version of CapTie [13] ran under MS-DOS because it was based on the header trace mode of OC3MON [1], [2].)

CapTie is a program running on a PC tapped at an ATM link. It monitors traffic on the link by using an ATM network interface card (NIC). CapTie can monitor the IPv4 traffic on ATM adaptation layer type 5 (AAL5). When the ATM NIC receives a cell and processes the physical layer, removes the LLC/SNAP encapsulation header and reads the packet length field in the packet header in the cell and generates a record consisting of the following four items: the CapTie counter value, the ATM NIC slot number, and the packet's timestamp and length. The record generated by CapTie is transferred to another PC that simulates (part of) a network using the record, and CapTie continues working (Figure 1). As shown later, the first target of the simulated network is the output buffers at the switching node in which the monitored link is accommodated.

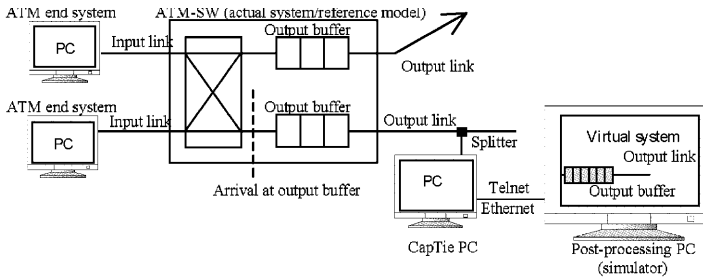


Fig. 1. ATM network with our system.

The record items are used as follows: (1) CapTie and the post-processing PC are connected via telnet over Ethernet. In our experience, the data rate between them is a few percent of the traffic on the monitored ATM link. Thus, when 100-Mbps traffic on the link is monitored, 1 Mbps or more of data traffic is transferred between the two PCs, so there is a danger of some data being lost. The value of CapTie's counter increases one by one when a packet is monitored. Thus, the post-processing PC can check whether any traffic records are lost. (2) CapTie can monitor more than one link simultaneously. To distinguish the record for the ATM link being monitored, an ATM NIC slot number is used. This allows network simulation to be performed if the PC running CapTie has many slots accommodating ATM NICs. (3) The timestamp is a key item of data for simulation. The timestamp uses 64 bits: 32 bits show the number of seconds and another 32 bits express the fraction of a second; this achieves nano-second granularity. (4) Packet length is another key item of data in the simulation. It determines the amount of traffic.

### 3 Estimation of an Arrival Sequence

The first goal of our traffic measuring tool is to reproduce the cell/packet arrival sequence at the transmission queue (output buffer) of the output link. (Note that CapTie monitors the cell/packet carried sequence of the link.) If this reproduction succeeds, we can analyze the traffic offered to the transmission queue accurately, simulate the transmission queue behavior, and can evaluate some interesting metrics. Otherwise, the result of the traffic analysis is misleading even when the timestamp has fine granularity. (For example, the peak traffic never exceeds the link capacity when we observe the traffic transmitted at the link.) However, the accurate reproduction of the cell/packet arrival sequence is not a straightforward task. Our measurement device is tapped at a link, so what we observe is not the cell arrival sequence or the packet arrival sequence but the cell transmission sequence or the packet transmission sequence. Therefore, our measuring device incorporates an algorithm for estimating the arrival sequence from the transmission sequence.

In this section, we investigate algorithms for estimating an arrival sequence from a transmission sequence. Here, the arrival sequence is defined by the series

of cell/packet arrival epochs observed just before the output transmission queue in the ATM switching node that has an ingress end point of the monitored link in Figure 1. The output buffer in the ATM switching node is sometimes called the reference model when we would like to emphasize the comparison between the reference model and the virtual model, which is a model used in the simulation (Figure 1). In the remainder of this section, UBR (unspecified bit rate) on the monitored ATM link is implicitly assumed because UBR is used for data transmission on ATM in most cases. As a result, cells of a packet are assumed to be offered to the network at the line speed, while packet transmission from an end system may be controlled by an upper layer.

### 3.1 Simple Cell Arrival Sequence Estimation Algorithm

Assume that CapTie observes the sequence of cells passing a tapping point in the middle of a link and judges whether a cell is the first cell of a packet based on the payload type indicator of the cell header. If it is the first cell of a packet, CapTie reads the packet length field of the IP packet header in the cell and generates a record consisting of the packet length and the timestamp showing when the cell was detected by CapTie. Actually we can observe the time instance when the first cell of each packet passes a certain point of a link and then observe how many cells belong to the same packet and will pass the point. Based on this information, the simple cell arrival sequence estimation (S-CASE) algorithm estimates the arrival instance of each cell at the arrival observation point.

Let  $t(i)$  be the observed timestamp of the first cell of the  $i$ -th packet, and let  $n(i)$  be the number of cells belonging to the  $i$ -th packet. Let  $H$  be the capacity of the input link to the ATM switching node that has the ingress end point of the monitored link (Figure 1) where we assume for simplicity that each input link has the same link capacity. Let  $L$  be the length of a cell. The following estimated arrival epoch of the  $j$ -th cell belonging to the  $i$ -th packet,  $t(i, j)$ , is provided by the S-CASE algorithm.

$$t(i, j) = t(i) + (j - 1) * L/H \quad (1)$$

The S-CASE algorithm assumes implicitly that the first cell does not wait for any time in the transmission queue and that the remaining cells arrive at the input link speed without interruption and do not wait in the transmission queue.

### 3.2 Busy Period Sensing Cell Arrival Sequence Estimation Algorithm

Assume that we can observe whether or not the output link is busy as well as observing the timestamp of each cell of a packet and the number of cells belonging to the packet. Let  $b$  be a busy flag, which is 1 when the output link is busy and 0 when it is idle. This flag can be obtained by directly observing the status of the output link or by checking the timestamp of each cell.

The busy period sensing cell arrival sequence estimation (BPS-CASE) algorithm estimates the arrival epoch of the  $j$ -th cell belonging to the  $i$ -th packet

as follows. It maintains  $x$ , the number of active flows on the output link, and  $T$ , the reference time. (i) The number  $x$  of active flows increases by one when the first cell of a packet is detected and decreases by one when the last cell of a packet is detected. That is, the definition of the number of active flows in this paper is the number of simultaneously existing packets on the link, and it is not defined by using the source (destination) address or source (destination) port number. (ii) If the link is busy at start time  $s$ , the reference time  $T$  is updated to be  $s$ . When the first cell of a packet is detected and if the timestamp is  $t$ , the reference time  $T$  is updated to be  $t$ .

The BPS-CASE algorithm stores the values of  $b$ ,  $x$ , and  $T$  for estimating the arrival epoch of each cell of the  $i$ -th packet when the first cell of the packet is detected. Let  $b(i)$ ,  $x(i)$ , and  $T(i)$  be their values just before the detection of the first cell of the  $i$ -th packet. By using them, the BPS-CASE algorithm estimates the arrival epoch of each cell of the  $i$ -th packet when the last cell of the packet is detected.

Consider the case in which the first cell of the  $i$ -th packet is detected when the link is idle (i.e.,  $b(i) = 0$ ). In this case, this cell does not wait for transmission. Therefore,  $t(i, 1) = t(i)$ , where  $t(i, j)$  is the estimated arrival epoch of the  $j$ -th cell of the  $i$ -th packet and  $t(i)$  is the timestamp of the first cell of the  $i$ -th packet.

Next, consider the case in which the first cell of the  $i$ -th packet is transmitted during a busy period of the link and assume that the link is kept busy while cells from the first one of the  $(i - 1)$ -th packet through the first one of the  $i$ -th packet are detected (Figure 2). In this case,  $T(i)$  is the transmission time (equivalently, the time stamp) of the first cell of the  $(i - 1)$ -th packet. Because the link is busy between  $T(i)$  and  $t(i)$ , the number of cells between the first cell of the  $(i - 1)$ -th packet and the first cell of the  $i$ -th packet is given by  $(t(i) - T(i)) * C/L$ , where  $C$  is the output link capacity (Figure 3). Since the number of active flows is  $x(i)$ , the mean time taken for this number of cells to arrive at the transmission queue of the reference model is  $(t(i) - T(i))C/(Hx(i))$  if each active flow offers cells at the input link capacity  $H$  and the number of active flows does not change between  $T(i)$  and  $t(i)$ . Therefore, if the estimate  $t(i - 1, 1)$  of the arrival epoch of the first cell of the  $(i - 1)$ -th packet is accurate, the arrival epoch of the first cell of the  $i$ -th packet can be estimated as

$$t(i, 1) = t(i - 1, 1) + (t(i) - T(i))C/(Hx(i)). \quad (2)$$

Here we should note that the estimated arrival epoch  $t(i, 1)$  of the first cell of the  $i$ -th packet should be less than or equal to the transmission time  $t(i)$  of this cell. To keep this constraint  $t(i, 1) \leq t(i)$  for any  $T(i) = t(i - 1) \geq t(i - 1, 1)$ , we use  $\max(C, Hx(i))$  instead of  $Hx(i)$ . Consequently, we obtain the following estimation rule that is applicable to both  $C > Hx(i)$  and  $C \leq Hx(i)$ .

$$t(i, 1) = t(i - 1, 1) + (t(i) - T(i))C/\max(C, Hx(i)). \quad (3)$$

Finally, consider the case in which the first cell of the  $i$ -th packet is transmitted during a busy period and the first cell of the  $(i - 1)$ -th packet is not included in this busy period (Figure 4). In this case,  $T(i)$  is the transmission epoch of

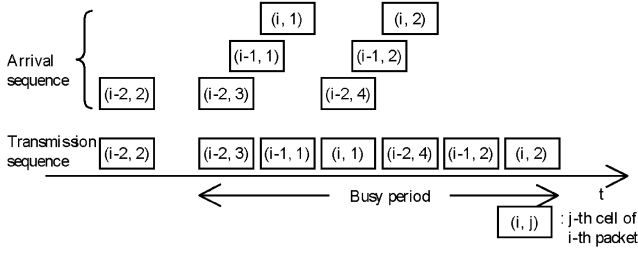


Fig. 2. Example of the event sequence (1).

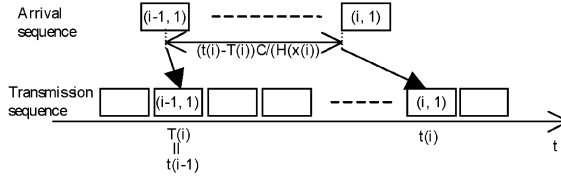


Fig. 3. Example of the event sequence (2).

the first cell forming this busy period. Note that the arrival epoch of this cell is  $T(i)$  because this cell does not wait for transmission. Therefore, similarly to the discussion above,

$$t(i, 1) = T(i) + (t(i) - T(i))C/\max(C, Hx(i)). \quad (4)$$

The arrival epoch of the  $j$ -th cell of the  $i$ -th packet is estimated as follows for any of the cases mentioned above.

$$t(i, j) = t(i, 1) + (j - 1)L/H \quad (5)$$

### 3.3 Numerical Examples

We investigated the accuracy of the S-CASE and BPS-CASE algorithms through a computer simulation, which simulated the whole system including the reference model and the point where the timestamp is applied (Figure 5). In this computer simulation, packets were generated according to a Poisson process and

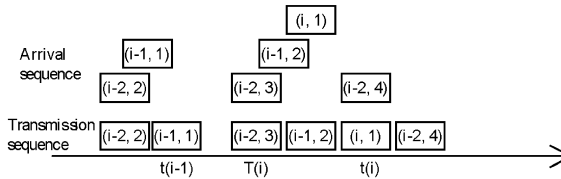
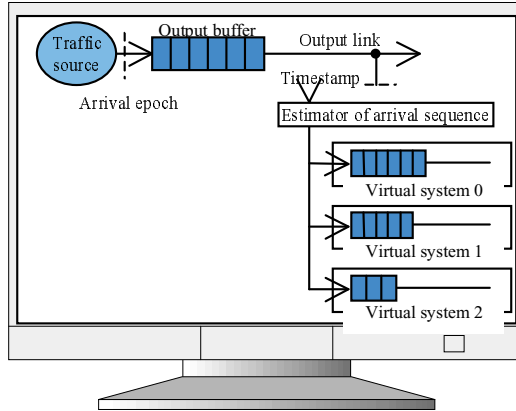


Fig. 4. Example of the event sequence (3).



**Fig. 5.** Computer simulation.

their length distribution was assumed to be geometric. They were offered to the transmission queue of the reference model (a FIFO queue). Through observation of the cell transmission sequence, the cell arrival sequence to the transmission queue of the reference model was estimated by the S-CASE and BPS-CASE algorithms. Since we knew the true cell arrival sequence in this computer simulation, we could compare the true and estimated cell arrival sequences. The accuracy was compared in terms of the difference in cell loss ratios (CLRs) that occurred when the true and estimated cell arrival sequences were offered to the virtual system and the reference model. This is explained below. One reason for measuring the accuracy in terms of the CLR is that the final target of the cell arrival sequence estimation is to use it for performance analysis.

By using the actual and estimated cell arrival sequences, the CLRs of the reference model and virtual systems in the computer simulation were evaluated, where the virtual systems were defined as a FIFO queue with a different buffer size or a different output link capacity from the reference model. In the following figures,  $(k, m, n)$  denotes the  $k$ -th simulation condition, the  $m$ -th virtual system, and the  $n$ -th estimation method. Simulation conditions and virtual systems (their buffer size and output link capacity) are summarized in Table 1. (The 0-th virtual system means the reference model.) The estimation method is 0 when the estimated cell arrival sequence is the true cell arrival sequence; 1 when the estimation method is the S-CASE algorithm; and 2 when it is the BPS-CASE algorithm. Under the following numerical examples, we assume that the average offer load from a source is 1 Mbps and that the number of sources is 30. (Thus, the total offered load is fixed.)

The results for Condition 1 are plotted in Figure 6. For high CLR or small output link capacity, both algorithms were accurate. As the CLR became lower or the output link capacity became larger, the S-CASE underestimated the CLR.

Figure 7 shows our investigation of Conditions 1 and 3. While the CLRs for the true cell arrival sequence were almost the same for  $(3, 0, 0)$  and  $(1, 2, 0)$ ,

**Table 1.** Simulation conditions and virtual systems

Simulation condition	Average packet length (Cell)	Input speed (Mbps)	Virtual system (buffer size, output link capacity) (Mbps)		
			0	1	2
1	10	50	(128, 50)	(128, 37)	(128, 30)
2	10	150	(128, 30)	(128, 37)	(128, 50)
3	50	50	(128, 50)	(128, 37)	(128, 30)
4	50	15	(128, 50)	(128, 37)	(128, 30)
5	10	150	(128, 50)	(128, 37)	(128, 30)
6	50	50	(256, 50)	(256, 37)	(256, 30)
7	50	50	(256, 50)	(128, 50)	(384, 50)
8	50	50	(128, 50)	(128, 75)	(128, 37)

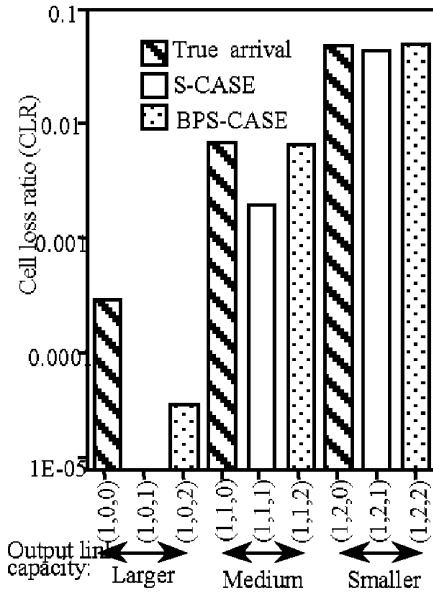
the estimation accuracies were different. Using either estimation algorithm, the CLR<sub>s</sub> in (1, 2, 1) and (1, 2, 2) were similar to the true CLR of (1, 2, 0). This means that the estimation in (1, 2, \*) was easy. On the other hand, the CLR<sub>s</sub> in (3, 0, 1) and (3, 0, 2) were different from the true CLR of (3, 0, 0). This means that the estimation in (3, 0, \*) was difficult. This seems to be mainly because it was easier to estimate the CLR of a small output link capacity (with short packets) than that of a large one (with long packets). (Item (v) below shows that long packets made estimation easier. Thus, a small output link capacity seems to be one of the main reasons for this result.)

Figure 8 compares the CLR<sub>s</sub> derived by the estimated and true cell arrival sequences. In addition, the CLR evaluated by the upper bound formula [4], [5] using the mean and peak cell rates of each flow is also shown for comparison. Here, the mean cell rate was the true mean cell rate and the peak cell rate was the cell rate transmitted by the input link capacity. In this figure, the  $y$ -axis denotes the log of the ratio of an estimated CLR to the true CLR. For example, for each simulation condition, S-CASE ((\*, 0, 1)/(\*, 0, 0)) means the log of the ratio of the CLR estimated by S-CASE for the virtual system 0 to the true CLR for the virtual system 0. That is, the  $y$ -axis denotes a metric of CLR estimation error.

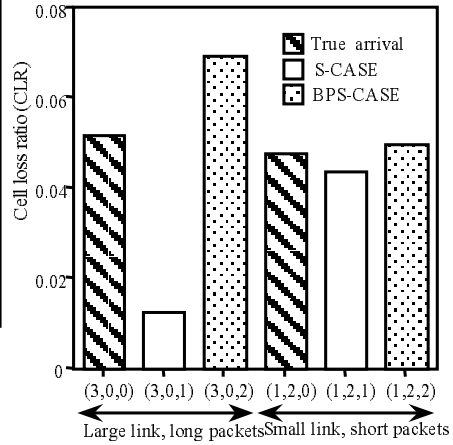
(i) We should note that the estimation was accurate when the capacity of the output link of the virtual system was smaller than that of the reference model, but may be inaccurate when it is larger. For example, for simulation condition 1, the estimation error for virtual system 0 when S-CASE was used (that is, S-CASE ((\*, 0, 1)/(\*, 0, 0))) was a larger absolute value than the estimation error for virtual system 1 when S-CASE was used (S-CASE ((\*, 1, 1)/(\*, 1, 0))). Here, virtual system 1 had less capacity than virtual system 0 (the reference model).

(ii) BPS-CASE is normally more accurate than S-CASE. For example, for simulation condition 1, S-CASE's estimation error for virtual system 1 (S-CASE ((\*, 1, 1)/(\*, 1, 0))) had a larger absolute value than BPS-CASE's (BPS-CASE ((\*, 1, 2)/(\*, 1, 0))).





**Fig. 6.** Cell loss ratio for condition 1.



**Fig. 7.** Errors in estimation for similar CLR.

(iii) Simulation conditions 1 and 5 for virtual system 0 and simulation condition 2 for virtual system 1 produced low CLR and the estimation was inaccurate with S-CASE for simulation conditions 1 and 5 and with BPS-CASE for simulation condition 2. See S-CASE  $((*, 0, 1)/(*, 0, 0))$ , S-CASE  $((*, 1, 1)/(*, 1, 0))$  and BPS-CASE  $((*, 1, 2)/(*, 1, 0))$ . It was difficult to estimate a low CLR accurately.

(iv) The accuracy of BPS-CASE deteriorated when the input link capacity was much larger than the output link capacity of the reference model. See BPS-CASE  $((*, 1, 2)/(*, 1, 0))$  for simulation condition 2 and compare it with that for simulation condition 1.

(v) BPS-CASE was accurate for simulation conditions 3, 4, and 6. There, the input link capacity was less than or equal to the output link capacity of the reference model and the packet size was long. It was more accurate for simulation condition 6 than for simulation condition 3, while the CLR for simulation condition 6 was lower than for simulation condition 3. (Normally, the estimation error becomes large when the CLR is lower.) This seems to be because the virtual system for simulation condition 6 had a longer buffer than that for simulation condition 3.

(vi) The absolute value of the estimation error obtained by BPS-CASE was always smaller (i.e., more accurate) than that by the upper bound formula.

(vii) We also compared the CLR of each virtual system for each simulation condition derived using the Poisson arrival assumption instead of the arrival sequence estimated by S-CASE or BPS-CASE. Here, the Poisson arrival assumption means that the cells were generated according to a Poisson process

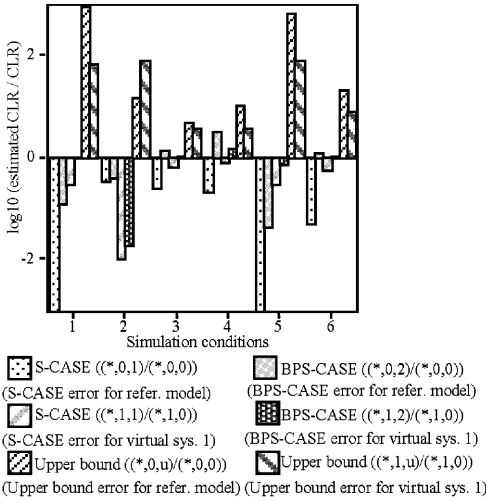


Fig. 8. Errors in estimation.

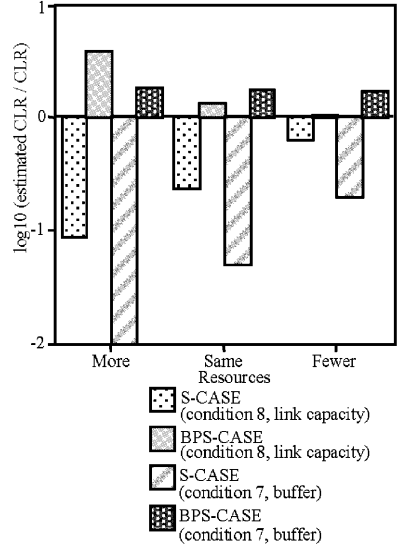


Fig. 9. Error in estimation for different network resources.

with mean equal to the mean observed in the simulation. The CLR obtained under the Poisson arrival assumption was smaller than -10 in this figure for all simulation conditions, so it was not plotted.

Figure 9 shows our investigation of conditions with different network resources (buffer and link capacity). When there were more network resources in the virtual system than in the actual system, the estimation errors were larger than when there were fewer network resources in the virtual system. BPS-CASE was more accurate and better than S-CASE. In particular, BPS-CASE was accurate even when the buffer size of the virtual system was different from that of the actual system.

## 4 Real-Time Simulation

The post-processing PC, which receives the data from CapTie via Ethernet, uses one of the arrival sequence algorithms mentioned in the previous section. As a result, we can (approximately) reproduce the cell arrival sequence at the multiplexing point (the transmission queue) of a switching node. Therefore, if we provide a simulator in the post-processing PC, provide a virtual system or a model of the transmission queue in the simulator, and give it the reproduced arrival sequence, we can simulate the performance behavior of new controls and new network resource conditions as if they were applied at the transmission queue (Figure 1).

In this study, we used a simulator to evaluate the accuracy of an output link capacity dimensioning method called the queue decay parameter method [13]. The simulator in the post-processing PC simulates an output link and its transmission queue in the switching node that has an ingress end-point of the monitored link. We assumed that the transmission queue could be modeled as a single-server FIFO queue with a finite-size waiting buffer where the service was cell transmission.

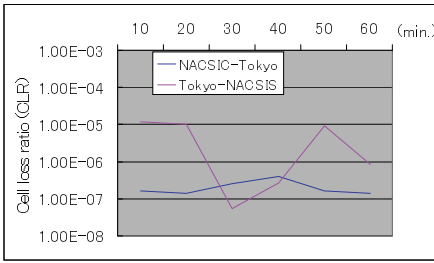
As a first step, the output link capacity was dimensioned by the dimensioning algorithm, which was proposed in [13] and used the traffic measurement data obtained by the switching node. The dimensioned output link capacity was used for the output link capacity (that is, the service rate of the server) in the simulator and the actual buffer size of the simulated transmission queue was used as the buffer size of the simulator. The estimated and reproduced cell arrival sequences were offered to the simulator in the post-processing PC, and the CLR of the transmission queue (output buffer) of the output link was evaluated in the simulator. In the numerical example below, S-CASE was used in the simulation for simplicity.

## 5 Numerical Example of Real-Time Simulation

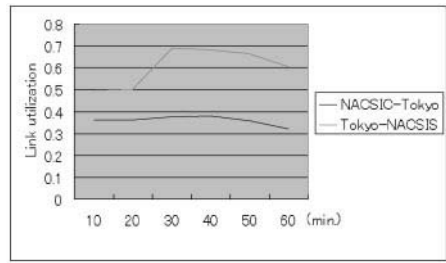
The developed system was applied to a bi-directional link in SINET (the Science Information Network) [3], which is a nationwide large IP network for research organizations in Japan, whose core network is implemented on an ATM network. The monitored link was between the University of Tokyo and the network office of NACSIS (National Center for Science and Information Systems). The capacity of each link was dimensioned using the queue decay parameter method [13] based on data measured in the week before this trial by an ATM switching node accommodating the link. The CLR objective was 10<sup>-6</sup>.

Figure 10 [13] plots the CLR of each link simulated by the real-time simulator for one hour during a busy hour. The CLR obtained by the simulator agreed closely with the objective. (In our experience, the CLR is a very difficult parameter to manage, so the agreement was actually far better than we expected.)

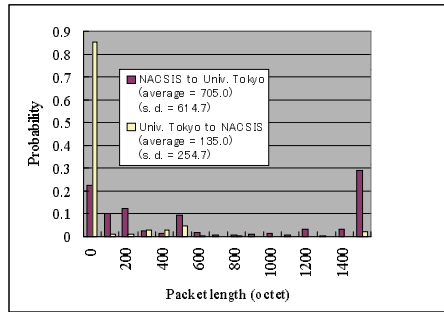
The direction from the NACSIS network office to the University of Tokyo has a large amount of traffic with long packets because it includes a lot of traffic downloaded from the U.S. The opposite direction has less traffic with shorter packets. The link utilization obtained in the simulation was higher in the direction from the University of Tokyo to the NACSIS than in the opposite direction (Figure 11). That is, the smaller link had higher utilization than the larger one whereas a larger link can usually achieve higher utilization due to the statistical multiplexing gain. The reason for this unexpected result is that short packets are dominant in the direction from the University of Tokyo to the NACSIS network office (Figure 12). However, the CLR of this direction had a larger fluctuation than that of the opposite direction because the offered traffic was small (Figure 10).



**Fig. 10.** Cell loss ratio in simulated out-put link.



**Fig. 11.** Link utilization in simulated output link.



**Fig. 12.** Packet length distribution.

Comparing Figures 10 and 11, we see that high average utilization during ten minutes does not mean a high CLR. This is because traffic fluctuations over a period much shorter than ten minutes strongly affect the CLR. Thus, we cannot estimate the CLR based only on a ten-minute average utilization.

## 6 Conclusions

We have developed a traffic monitoring tool called CapTie and a real-time simulator that uses data from CapTie. The cell arrival sequence estimation methods used in it were evaluated. Using these two systems, the proposed dimensioning method was evaluated for real IP traffic data on an ATM network and shown to be accurate. The developed system enables us to implement a new approach to network engineering. In the near future, we will apply the system to a self-sizing network.

## References

1. K. Thompson, et al., IEEE Network, pp. 10-23, Nov./Dec. (1997).
2. <http://www.nlanr.net/NA/Oc3mon>

3. S. Asano, IEICE Journal, 81, 4, pp. 402-406 (1998).
4. H. Saito, Teletraffic Technologies in ATM Networks, Artech House, Boston (1994).
5. H. Saito, IEEE Trans. Communications, 40, 9, pp. 1512-1521 (1992).
6. N. G. Duffield, et al., Proc. Cam. Phil. Soc., 118:363-374 (1994).
7. H. Saito, et al., NTT Review, 8, 1, pp. 56-65 (1996).
8. S. Nakagawa, et al., NOMS'96, Kyoto (1996).
9. H. Saito, IEEE Communication Magazine, 35, 5, pp. 146-153 (1997).
10. <http://www.wrs.com/products/html/vxworks.html>
11. M. Maeda, IEEE ATM workshop '99, Kochi (1999).
12. <http://www.tmo.hp.com/tmo/ntd/products/products.html>
13. H. Saito, et al., ICCCN'99, Boston (1999).

# Computing Stochastic Bounds for the Tail Distribution of an M/GI/1 Queue

Pierre L. Douillet<sup>1</sup>, André-Luc Beylot<sup>2</sup>, and Monique Becker<sup>3</sup>

<sup>1</sup> Mathématiques Supérieures, CPGE Faidherbe,  
9, rue A. Carrel, 59000 Lille, France  
`douillet@cnam.fr`

`http://www-inf.int-evry.fr/ douillet/`  
<sup>2</sup> Laboratoire PRiSM, Université de Versailles,  
78035 Versailles Cedex -France  
`beylot@prism.uvsq.fr`

<sup>3</sup> Institut National des Télécommunications,  
rue Charles Fourier 91011 Evry Cedex, France  
`mbecker@etna.int-evry.fr`

**Abstract.** Packet switching networks lead mostly to M/GI/1 queue models. In this paper, computing methods are designed in order to get quickly approximate values for response time of these networks. Laplace transform is a powerful tool to study such queuing systems. But inversion of the Laplace transform on the real line is well known to be an ill-conditioned problem and usual numerical methods of inversion fail to give accurate error bounds.

A new method to address this old problem is afforded by the recently developed formal computing tools: exact computations can be done during the first steps of calculation, while usual floating point computations remain confined to the last steps. Applying that method to an M/GI/1 queue, a formal approach is designed, leading to proven bounds, and several numerical improvements are proposed. Accurate bounds are obtained.

## Introduction

The M/GI/1 queue is the most useful model for packet switching networks. It is often realistic to assume that packet arrival process is Poisson, but service time is not exponential at all. Services are independent when treatments depend upon the packet length, and when this length is constant, the adequate model is M/D/1. Otherwise M/GI/1 queues are used.

In this paper are considered known results [1] about M/GI/1 queue, but from a new point of view, allowed by an efficient use of formal computing. Efficient algorithms are designed in order to obtain bounds on performance criteria for such a queue. The tail of the response time distribution of such an M/GI/1 queue can be evaluated from Pollaczek-Khintchine formula, leading to deal with Laplace transforms.

But when dimensioning systems, Laplace transforms have to be inverted, and the usual numerical methods give approximations for which accurate error bounds are not available. This is rather not a new subject [2][3][4][5], but a new method to attack this old problem can be obtained as follows.

We have written, in Maple code, a package of procedures that computes the formal convolution of two piecewise polynomial functions (i.e. splines). This is especially usefull when dealing with inverse Laplace transform, since such a computation is strongly ill-conditioned. A method using exact computations during the first steps of iterative algorithms is therefore significantly more accurate than any other method. From these formal computations, two kinds of results concerning the sojourn-time pdf can be obtained: exact bounds for the first part of the curve and an accurate approximation of the second part of the curve. Moreover, unlike other algorithms, the proposed algorithm leads to an error term that is not an oscillating function, and requires no "calibrations".

The present paper is organized as follows: assumptions and notations are presented in section 1. The inversion of the Pollaczek-Khintchine formula is reminded in section 2. Two formal approaches are designed in sections 3 and 4. Numerical improvements are proposed in section 5. Conclusions are formulated in section 6, and the paper ends with some references.

## 1 Assumptions and Notations

Let us use the following notations. When  $\mathbf{X}$  is some random variable, its pdf (probability density function) will be noted  $x$ , its cdf (cumulative distribution function) will be noted  $X$  and its Laplace transform will be noted  $\hat{x}$ . In other words  $x(u) du = Pr \{ \mathbf{X} \in [u, u + du] \}$ ,  $X(u) = \int_0^u x(t) dt$ , and  $\hat{x}(z) = \int_0^\infty x(t) \exp(-zt) dt$ .

Let  $\mathbf{A}$  be the random variable "inter-arrival duration" and  $\lambda = 1/\mathbf{E}(\mathbf{A})$ , so that:  $a(t) = \lambda \exp(-\lambda t)$ . Let  $\mathbf{B}$  be the random variable "service duration" and  $\mu = 1/\mathbf{E}(\mathbf{B})$ . This last value is the average throughput *during busy periods*, and the quantity  $\rho = \lambda/\mu$  is the load of the system. It is assumed that no infinite queue happens (therefore  $\rho < 1$ ).

The random variable  $\mathbf{R}$  will denote the sojourn time of a customer in the system,  $\mathbf{W}_0$  its (perhaps being null) waiting time, and  $\mathbf{W}_1$  its conditional waiting time, i.e. the waiting time knowing that the customer will wait. All the preceding random variables, and especially the service duration  $\mathbf{B}$ , are "individual variables", i.e. can be perceived as the result of a process that picks a customer by uniform sorting over their order of arrival, and notes the value associated to this customer.

Another process of selection can be used, that picks a customer by uniform sorting over the instants of observation (that sorting being repeated until a client is eventually served), leading to "temporal" variables. Let  $\beta$  be the pdf of  $\tilde{\mathbf{B}}$  (the temporal service duration). Relation  $\beta(t) = \mu t b(t)$  is obvious and gives  $\mathbf{E}(\tilde{\mathbf{B}}) = (1/\mu) + \mu \text{var}(\mathbf{B})$ , while the variance of  $\tilde{\mathbf{B}}$  includes the third moment of  $\mathbf{B}$ .

By definition, a customer arriving in a non-empty system finds another customer that is currently being served (and maybe other ones, being waiting). The duration that the incoming customer has to wait until the served customer leaves the system is called the "residual service time", and will be noted  $\mathbf{C}$ . The pdf of  $\mathbf{C}$  is well-known to be  $c(t) dt = \mu dt \int_{u=t}^{\infty} b(u) du$ , and we have  $\mathbf{E}(\mathbf{C}) = \frac{1}{2}\mathbf{E}(\tilde{\mathbf{B}})$ , while  $\mathbf{var}(\mathbf{C})$  involves up to the third moment of  $\mathbf{B}$ . In what follows, the notations  $\bar{x}$  and  $\sigma^2$  will *always* refer to  $\mathbf{E}(\mathbf{C})$  and  $\mathbf{var}(\mathbf{C})$ .

## 2 Inversion of the Pollaczek-Khintchine Formula

### 2.1 Introduction

It is well known that the Laplace transforms of the sojourn time  $\mathbf{R}$  and the service duration  $\mathbf{B}$  are related by the Pollaczek-Khintchine formula:  $\hat{r}(z) = \frac{(1-\rho)z\hat{b}(z)}{z-\rho\mu+\rho\mu\hat{b}(z)}$ . But the residual service time obeys to  $\hat{c}(z) = \mu \frac{1-\hat{b}(z)}{z}$ , and we have:

$$\hat{r}(z) = \hat{b}(z) \frac{(1-\rho)}{1-\rho\hat{c}(z)} \quad (1)$$

Since the Laplace transform of the sum of two independent r.v.'s is the product of the Laplace transforms of these r.v.'s, (1) leads to  $\widehat{w_0}(z) = \frac{1-\rho}{1-\rho\hat{c}(z)}$ , the sojourn time of a customer being the sum of his waiting time and his service duration.

The computation of  $\widehat{w_1}$  may be derived when noticing that a conditional waiting time is the sum of a residual service duration and an ordinary waiting time, leading to  $\widehat{w_1} = \hat{c}\widehat{w_0}$ . It may also be noticed that an ordinary waiting time is either null (with probability  $1-\rho$ ) or is a conditional waiting time (with probability  $\rho$ ), leading to  $\widehat{w_0} = (1-\rho) + \rho\widehat{w_1}$ , i.e. to the same result.

### 2.2 An Inverse Transform Method

The power-series expansion of (1) is:  $\hat{r}(z) = (1-\rho) \hat{b}(z) \sum_{k \geq 0} (\rho\hat{c}(z))^k$  and these series converge (when  $z$  is in the right-hand half-plane, i.e.  $\Re(z) \geq 0$ ) since  $c$  verifies:  $|\hat{c}(z)| = |\int_0^\infty c(t) \exp(-zt) dt| \leq |\int_0^\infty c(t) dt| = 1$  and  $\rho < 1$ . Therefore we have:

$$r(t) = b(t) \otimes \sum_{k \geq 0} (\rho^k c[k](t)) \Big/ \sum_{k \geq 0} \rho^k \quad (2)$$

where  $\otimes$  is the convolution operator,  $c[k](t)$  is the convolution of  $k$  functions equal to  $c(t)$  and therefore  $c[0](t) = \text{Dirac}(t)$ . That formula can be rewritten as:

$$r(t) = (1-\rho)b(t) + \rho b(t) \otimes w_1(t) \quad (3)$$

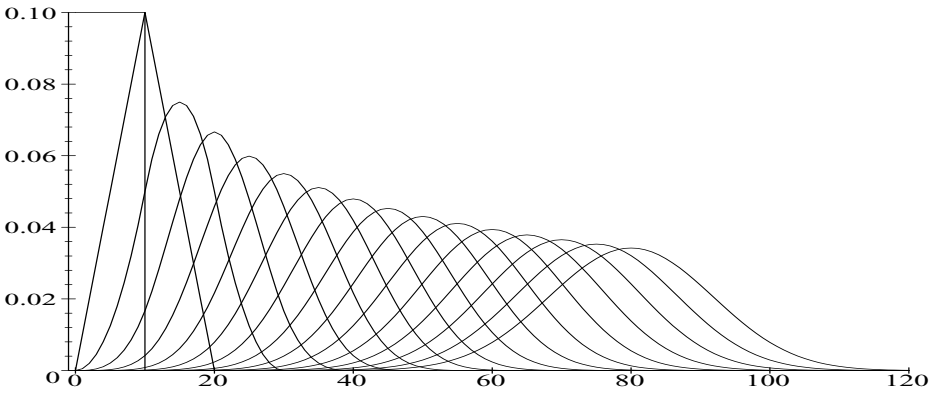
$$w_1(t) = \sum_{k \geq 1} (\rho^k c[k](t)) \Big/ \sum_{k \geq 1} \rho^k \quad (4)$$



Equation (3) says again that the sojourn time is equal to the service time when a customer arrives in an empty queue (which happens with probability  $1 - \rho$ ) and otherwise is increased by the conditional waiting time. Equation (4) says that the pdf of the conditional waiting time may be obtained by a disjunction into an infinite number of cases, each one occurring with probability  $(1 - \rho) \rho^{k-1}$  and corresponding to the sum of  $k$  independent residual service times [7]. Let us note  $r[n]$  and  $w_1[n]$  the functions obtained by truncating the summations in both numerator and denominator of (2) and (4). These functions have again a unit mass, i.e. are again the pdf of some random variables.

### 2.3 Example of Deterministic Services

Let us consider a deterministic server with  $\rho = 0.75$ ,  $\mu = 0.1$ . In other words,  $c(t) = \mu = 0.1$  when  $t < 10$  and  $c(t) = 0$  otherwise. After some computations [8], the convolutions  $c[2] \dots c[16]$  of the residual service time pdf can be obtained, leading to Fig. 1.



**Fig. 1.** M/D/1 file: from  $c[2]$  to  $c[16]$ .

These functions become more and more regular, while flattening on the horizontal axis and sliding towards right. This sliding is due to  $\mathbf{E}(c[n]) = n \mathbf{E}(\mathbf{C}) = n \bar{x}$ , while flattening is due to  $\mathbf{var}(c[n]) = n \mathbf{var}(\mathbf{C}) = n \sigma^2$ : when the curve gets wider, its height has to decrease, since the area underneath the curve is constant. Getting regular is nothing but the central limit theorem, i.e. the convergence of the reduced pdf of a sum of i.i.d. variables towards the normal law.

Figure 2 shows the approximations  $w_1[3] \dots w_1[16]$  of the conditional waiting time distribution. It appears that those functions converge to a limit function and that convergence is faster for small values of  $t$ . More precise statements about this convergence will be given in next section.

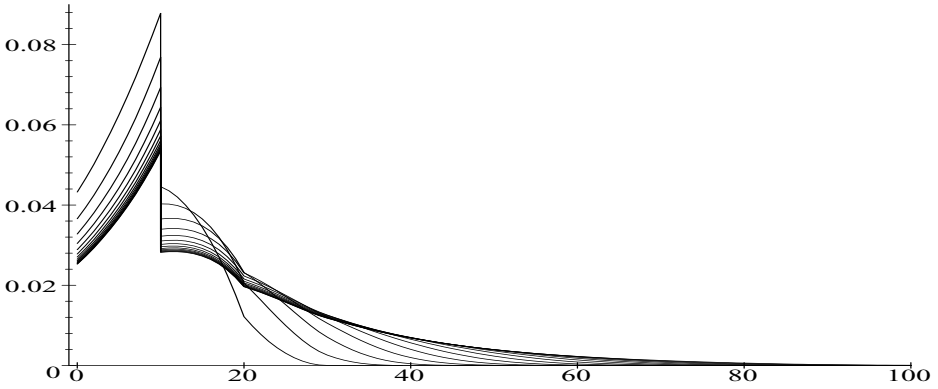


Fig. 2. M/D/1 file: from  $w_1$  [3] to  $w_1$  [13].

### 3 First Bounds

#### 3.1 Statement and Illustration

According to our general notations,  $R[n](t)$  will denote  $\int_0^t r[n](u) du$ . A first claim is:

$$\forall n : (1 - \rho^{n+1}) R[n] \leq (1 - \rho^{n+2}) R[n+1] \leq R \leq R[n+1] \leq R[n] \quad (5)$$

As an illustration of this theorem, the left part of Fig. 3 shows the sequence of the lower bounds relative to and for an M/D/1 queue where the right part shows the related upper bounds.

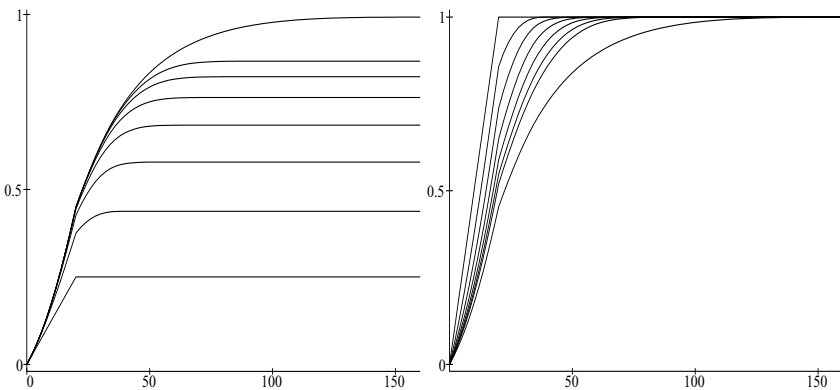


Fig. 3. M/D/1 file: cdf lower (left) and upper (right) bounds

### 3.2 Proof

For the left part: from the very definition of  $r[n]$ ,  $(1 - \rho^{n+1}) r[n](t)$  equals  $\sum_{n \geq k} \rho^k b(t) \otimes c[k](t)$ . Convolution products of  $c$  and  $b$  are pdf's, so every term in this sum is positive and the sequence of functions  $(1 - \rho^{n+1}) r[n](t)$  is increasing. This remains true when integrating, so the sequence of functions  $(1 - \rho^{n+1}) R[n](t)$  is also increasing.

For the right part: each  $c[k]$  is the pdf of a r.v.  $\mathbf{X}_k$ , equal to the sum of  $k$  i.i.d. variables  $\mathbf{X}_{1,k} \dots \mathbf{X}_{k,k}$ , each one being distributed according to  $c$ . The function  $w_0[n]$  can be viewed as the pdf of the r.v.  $\mathbf{Y}_n$  defined by: pick a  $k \in [0, n]$  according to weight  $\rho^k$  (i.e. with a probability proportional to  $\rho^k$ ) and then pick a  $\mathbf{X}_k$ . Therefore,  $1 - R[n](t)$  is the probability that  $\mathbf{B} + \mathbf{Y}_n \geq t$  while  $1 - R[n+1](t)$  is the probability that  $\mathbf{B} + \mathbf{Y}_{n+1} \geq t$ . Since the odds of the former are obviously lower than those of the latter, the sequence  $R[n]$  is decreasing.

### 3.3 Limitations of This Result

The bounds given in (5) are simple, and apply to the whole scale of response times. But their efficiency has to face two kinds of remarks. Firstly, and it can be easily seen on Fig. 3, the exact curve is not at the same distance from the two bounds, starting very near to the lower bound and ending very near to the upper bound, giving the limit value  $R(\infty) = 1$ .

Secondly, these bounds provide nearly no information about the tail of the distribution. Let us for example consider a deterministic law. It may easily be seen that residual service duration pdf is:  $c(t) = \mu \chi_{[0; 1/\mu]}$  where  $\chi$  is the usual indicator function (1 inside, 0 outside). Therefore  $c[k]$  is zero outside  $[0; k/\mu]$  and  $R[k](t) = 1$  when  $t \geq k/\mu$ . So let us look for more efficient bounds.

## 4 Better Bounds

### 4.1 Statement and Illustration

Let us now prove:

$$\forall n : (1 - \rho^{n+1}) w_1[n] \leq (1 - \rho^{n+2}) w_1[n+1] \leq w_1 \quad (6)$$

$$\forall n : \exists T_n : \forall t \leq T_n : w_1(t) \leq w_1[n+1](t) \leq w_1[n](t) \quad (7)$$

As an illustration of this theorem, Fig. 4 (left) shows the sequence of the lower bounds relative to  $n = 1..8$  and  $n = 16$  for an M/D/1 queue (i.e. for a deterministic service distribution) where  $\rho = 0.75$ ;  $\mu = 0.1$ , while Fig. 4 (right) shows some  $T_n$  values.

Inequality (6) may be proved in the same way as the left hand inequality in (5). It leads to a global lower bound of the conditional waiting time pdf. The only existence of a  $T_n$  satisfying inequality (7) is trivial ( $T_n = 0$  works !): what is to be proved is that an optimal value can be derived and that the corresponding  $T_n$  is meaningful.

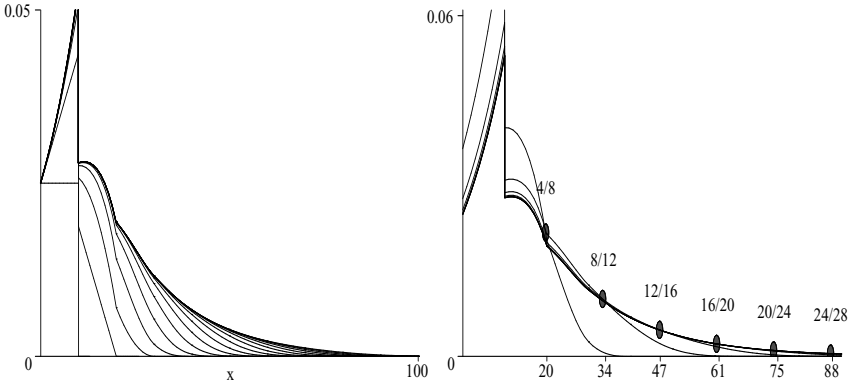


Fig. 4. Left: lower bounds. Right : upper bounds.

## 4.2 Asymptotic Approximation

Let us begin by computing the coordinate  $T_{n,n+j}$  where the two curves  $w_1[n]$  and  $w_1[n+j]$  intersect. Figure 5 plots  $T_{n,n+1}$  and  $T_{n,n+4}$  against  $n$ : the resulting graph appears as almost rectilinear. Computing the regression line (M/GI/1 file, with the given values of parameters) leads to  $T_{n,n+1} \approx 3.468 + 3.401n$  and  $T_{n,n+4} \approx 6.032 + 3.406n$ . Each of these regression lines is "strongly explicative", since it reduces the standard deviation of the  $T_n$ 's by a factor 300.

Observing that  $x \geq T_{n,n+j}$  induces  $w_1[n](x) \geq w_1[n+j](x) \geq w_1(x)$ , these regression lines can be used to choose the number of requested iterations to obtain reliable bounds on a given initial interval. With the parameters' values that were chosen in the example, it appears that exact bounds on the  $[0, 200]$  interval are obtained when  $n = 57$ . Using this value leads to a relative error less than  $\rho^{57} \approx 10^{-6}$ . Therefore, events whose probability is more than  $10^{-5}$  are almost exactly described (this probability corresponds to  $x = 200$ ).

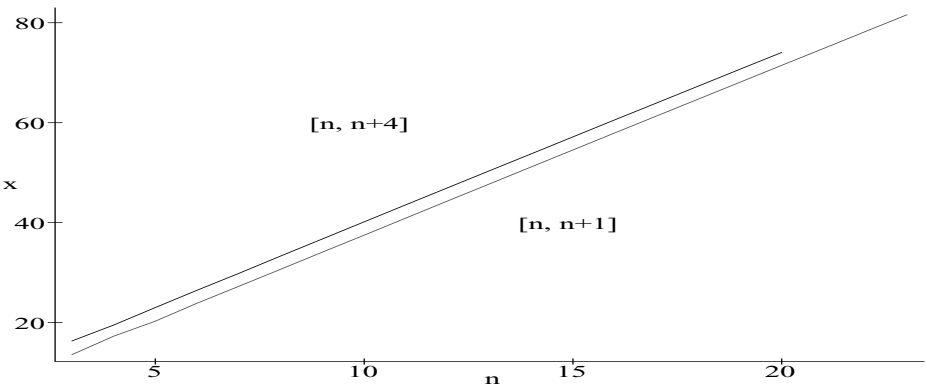
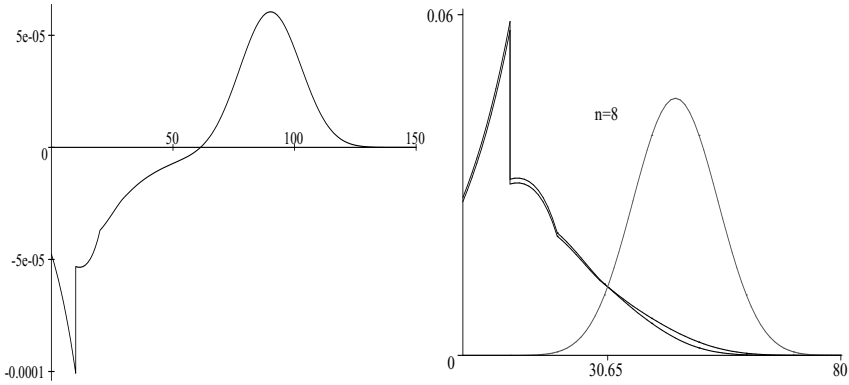


Fig. 5. Plotting  $T_{n,n+1}$  and  $T_{n,n+4}$  against  $n$ .



**Fig. 6.** Left:  $w_1[18] - w_1[17]$ . Right :  $w_1[8]$ ,  $w_1[9]$  and  $c[9]$ .

### 4.3 Proof

To prove the existence and unicity of an abscissa  $T_{n,n+1}$  such that  $w_1[n](x) = w_1[n+j](x)$ , the main point is to observe that the difference between two successive  $w_1[n]$  functions goes from negative values to positive values only once (Fig. 6, left).

This situation can be derived from the fact that  $w_1[n+1] - w_1[n]$  is proportional to  $c[n+1] - w_1[n]$ , and therefore we are intersecting  $w_1[n]$  and  $c[n+1]$ . By the way (Fig. 6, right), we obtain a better conditioned numerical equation. More precisely,

$$w_1[n+1](t) - w_1[n](t) = \frac{(1-\rho)\rho^n}{1-\rho^{(n+1)}} (c[n+1](t) - w_1[n](t))$$

For increasing values of  $n$ ,  $c[n]$  becomes indistinguishable from  $n\text{law}(\bar{x}n, \sigma\sqrt{n})$ , the normal law with mean  $n\mathbf{E}(\mathbf{C}) = n\bar{x}$  and variance  $n\mathbf{var}(\mathbf{C}) = n\sigma^2$ , while  $w_1[n]$  converges towards  $w_1$ . But, for sufficiently large  $t$ ,  $w_1$  is indistinguishable from an exponential law (large deviation theorem). When  $c$  is zero outside some finite domain, we have the more precise result :  $w_1(t) \approx w_{\text{large dev}}(t)$  when  $t \rightarrow \infty$  where  $w_{\text{large dev}}(t) \doteq \sum_{k \geq 1} (\rho^k n\text{law}(\bar{x}n, \sigma\sqrt{n})(t)) / \sum_{k \geq 1} \rho^k$ . One obtains:

$$w_{\text{large dev}}(t) \approx \frac{1-\rho}{\rho} \frac{1}{\eta} \exp\left(-\frac{\eta - \bar{x}}{\sigma^2} t\right) \quad \text{where} \quad \eta = \sqrt{\bar{x}^2 - 2 \ln(\rho) \sigma^2} \quad (8)$$

In our example, numerical tests show that (8) is efficient as soon as  $t \geq 10\bar{x}$ .

Therefore, intersecting  $w_1[n]$  and  $w_1[n+1]$  is equivalent to intersect a fixed (exponential) curve decreasing to 0 and a moving to right (bell) curve increasing from 0, proving the existence and unicity of  $T_n, n+1$  and the relation  $T_{n+1, n+2} > T_{n, n+1}$ . Combining these relations, we obtain that, for a fixed  $n$ , the sequence  $T_{n, n+j}$  is increasing. Its limit is the requested  $T_n$ .

The core of the preceding proof being decreasing versus increasing (and not exponential versus normal), the result still holds for small values of  $n$  (without requiring a formal proof, since a simple examination of the curves is sufficient).

#### 4.4 About Complexity

The computation from convolution of functions  $r[n]$ ,  $R[n]$ ,  $c[n]$  has a cost, that needs to be evaluated carefully. For M/D/1 queues, the  $w_1[n]$  are piecewise polynomial functions ( $n + 1$  parts of degree  $n - 1$ ). For the first iterations, it is efficient to use fractional numbers, while for the next ones, it is necessary to use floating point numbers. Then it is necessary to use a sufficient precision in order to avoid instability problems due to inversion of Laplace transform [6].

We used up to 100 decimal digits (which let us get an accuracy of only 5 useful digits for the final result). Programming carefully and using recent computers allow to compute convolutions of piecewise functions up to 100 pieces, each of degree 100, but not more.

### 5 Two Improvements

#### 5.1 Smoothing and Fast Convolution

Let us see now two techniques giving a better approximation of  $w_1$  beyond  $T_n$ . First an approximate value of  $w_1[2n]$  may be obtained from  $w_1[n]$  by a fast exponential technique based on the following formula:

$$\sum_{k=1}^{k=2n} \rho^{k-1} c[k] = \sum_{k=1}^{k=n} \rho^{k-1} c[k] + \rho^n c[n] \otimes \sum_{k=1}^{k=n} \rho^{k-1} c[k] \quad (9)$$

Let us note that a direct application of this formula does not improve the computation complexity. On the contrary it increases it. Of course, there are fewer convolutions to compute, but each of them takes into account functions which have a larger number of pieces, made of polynomials with higher degree: the number of integrations is higher and so is computing time. For a computation based on formula (9) to be possible, smoothing of  $c[n]$  and  $w_1[n]$  functions is necessary so as to decrease both the degree and the number of pieces.

An experimental study shows that applying (9) to the functions obtained by smoothing an exactly known  $w_1[n]$  yields an important gain. But this method cannot be applied again, since the second smoothing, again needed to lower both degree and number of pieces, ruins the quality of approximation.

#### 5.2 Again, the Central Limit Theorem

Another technique for approximating  $w_1[2n]$  may be obtained as a consequence of central limit theorem:  $c[k]$  functions, when centered and reduced to a unit variance converge to a normal distribution, since a convolution of pdf's leads to

the pdf of the sum of r.v.'s. Therefore, an approximation of even indexed  $w_1$ 's is obtained by writing that  $w_1[n](n\bar{x} + \tau\sqrt{n}\sigma) \approx \sqrt{2}w_1[2n](2n\bar{x} + \tau\sqrt{2n}\sigma)$ .

It is in fact better to use  $w_1[n](n\bar{x} + \tau\sqrt{n}\sigma) \approx 2w_1[4n](4n\bar{x} + 2\tau\sqrt{n}\sigma)$  which does not introduce new linking points and does not increase the complexity of the next computations. Starting from  $n = 31$ ,  $w_1[32]$  may be approximated from  $w_1[8]$  and so on until  $w_1[124]$  from  $w_1[31]$ , which allows the computation of  $\sum_{j=8}^{31} \rho^{4j-1} c[4j]$ . Thereafter, a convolution with  $\sum_{j=1}^3 \rho^k c[k]$  allows the computation of the other terms (whose indices are not  $4n$ ).

## 6 Conclusion

In this paper, computing methods have been designed in order to evaluate response time of packet switching networks. Efficient algorithms have been proposed to study M/GI/1 queues. They use formal and thereafter numerical techniques. They decrease approximation errors for the set of values which are not small enough to use standard techniques and not large enough to use large deviation techniques. Examples were shown from M/D/1 queues, but the algorithms apply to the general case. A key point to conclude: the proposed method does not generate a Gibbs' phenomenon at discontinuity points (Fig. 2), unlike others algorithms based upon orthogonal functions, and the remaining error term is no longer an oscillating function.

## Acknowledgments

Thanks to the anonymous referees for helpful comments.

## References

1. 1959. Gaver D. P. "Embedded Markov chain analysis of a waiting line process in continuous time", *Annals Math. Stat.* 30, 698-720
2. 1968. (Dubner-Abate algorithm) : Dubner H. and J. Abate. "Numerical inversion of Laplace transforms by relating them to finite Fourier-cosine transform", *J. ACM* 15(1):115-123.
3. 1971. (Salzer-Piessens algorithm) : Piessens, R. "Gaussian quadrature formulas for the numerical integration of Bromwich's integral and the inversion of the Laplace transform". *J. Engineer. Math.* 5(1):1-9.
4. 1970. (Gaver-Stehfest algorithm) : Stehfest H. "Algorithm 368 [D5] : Numerical inversion of Laplace transforms", *Comm. ACM* 13(1):47-49 and 13(10):624.
5. 1966. (Widder-Weeks algorithm) : Weeks W.T. "Numerical inversion of Laplace transforms using Laguerre functions", *J. ACM.* 13(3):419-426.
6. 1994. Cheng, A.H-D., P. Sidaurik and Y. Abousleiman. "Approximate inversion of the Laplace transform", *The Mathematica Journal* 4(2):76-82.
7. 1975. Kleinrock. "Queueing systems, Vol. 1: theory", Wiley
8. 1998. Douillet P.L. "A study of some queueing systems", Ph.D. Thesis, University of Paris, 253 p. Available at : [www-inf.int-evry.fr/~douillet/thesis](http://www-inf.int-evry.fr/~douillet/thesis).

In [8], more than 200 references, are given concerning inverse Laplace Transform, queueing systems and formal calculus.

# Analysis of Packet Delay in a GI-G-1 Queue with Non-preemptive Priority Scheduling

Joris Walraevens, Bart Steyaert, and Herwig Bruneel

SMACS Research Group  
Ghent University, Vakgroep TELIN (TW07V)  
Sint-Pietersnieuwstraat 41, B-9000 Gent, Belgium.

Phone: 0032-9-2648902  
Fax: 0032-9-2644295  
{jw,bs,hb}@telin.rug.ac.be

**Abstract.** Priority scheduling for packets is becoming a hot topic, as attempts are being made to integrate voice services in existing IP data networks. In this paper, we consider a discrete-time queueing system with head-of-line (HOL) non-preemptive priority scheduling. Two classes of traffic will be considered, i.e., high priority and low priority traffic, which both generate variable-length packets. We will derive expressions for the Probability Generating Function (pgf) of the packet delay of the high priority traffic and the low priority traffic. From these, some performance measures (such as the mean value) will be derived. These will be used to illustrate the significance of priority scheduling and the effect of non-preemptive scheduling on the high priority traffic.

## 1 Introduction

In recent years, there has been much interest devoted to incorporating multimedia applications in IP networks. Different types of traffic need different QoS standards. For real-time applications, it is important that mean delay and delay-jitter are bounded, while for non real-time applications, the Loss Ratio (LR) is the restrictive quantity.

In general, one can distinguish two priority strategies, which will be referred to as Time Priority and Space Priority. Time priority schemes attempt to guarantee acceptable delay boundaries to delay-sensitive traffic (such as voice/video). This is achieved by giving it HOL priority over non-delay-sensitive traffic, and/or by sharing access to the server among the various traffic classes in such a way so that each can meet its own specific delay requirements. Several types of Time priority (or scheduling) schemes (such as Weighted-Round-Robin (WRR), Weighted-Fair-Queueing(WFQ)) have been proposed and analyzed, each with their own specific algorithmic and computational complexity (see e.g. [6] and the references therein). On the other hand, Space Priority schemes attempt to minimize the packet loss of loss-sensitive traffic (such as data). Again, various types of Space Priority (or discarding) strategies (such as Push-Out Buffer (POB), Partial Buffer Sharing (PBS)) have been presented in the literature (see



e.g. [15]), mainly in the context of ATM buffers. An overview of both types of priority schemes can be found in [1].

In the existing literature, there have been a number of contributions with respect to HOL priority scheduling. An overview of some basic HOL priority queueing models can be found in Jaiswal [3], Takacs [10] and Takagi [11] and the references therein. Khamisy et al. [4], Laevens et al. [5], Takine et al. [13] and Walraevens et al. [16] have studied discrete-time HOL priority queues with deterministic service times equal to one slot. Furthermore, non-preemptive HOL priority queues have been considered by Rubin et al. [7], Stanford [8], Sugahara et al. [9] and Takine et al. [12,14]. Rubin [7] studies the mean waiting time, for a queue fed by an i.i.d. arrival process. Stanford [8] analyses the interdeparture time distribution in a queue fed by a Poisson process. In Sugahara [9], a non-preemptive queue in continuous time is presented, with a Switched Poisson Process arrival process for the high priority packets. Finally, Takine [12,14] studies a discrete-time MAP/G/1 queue, using matrix-analytic techniques.

In this paper, we analyse the packet delay of high and low priority traffic in a discrete-time single-server buffer for a non-preemptive HOL priority scheme and per-slot i.i.d. arrivals. The transmission times of the packets generated by both types are assumed to be generally distributed. We will demonstrate that an analysis based on generating functions is extremely suitable for modelling this type of buffers with priority scheduling.

## 2 Mathematical Model

We consider a discrete-time single-server queueing system with infinite buffer space. Time is assumed to be slotted. There are 2 types of traffic arriving in the system, namely packets of class 1 and packets of class 2. We denote the number of arrivals of class  $j$  during slot  $k$  by  $a_{j,k}$  ( $j = 1, 2$ ). Both types of packet arrivals are assumed to be i.i.d. from slot-to-slot and are characterized by the joint probability mass function  $a(m, n)$  and joint probability generating function (pgf)  $A(z_1, z_2)$ . Notice that the number of packet arrivals from different classes (within a slot) can be correlated. Further, we define the marginal pgf's of the arrivals from class 1 and class 2 during a slot by  $A_1(z) \triangleq E[z^{a_{1,k}}] = A(z, 1)$  and  $A_2(z) \triangleq E[z^{a_{2,k}}] = A(1, z)$  respectively. We furthermore denote the arrival rate of class  $j$  ( $j = 1, 2$ ) by  $\lambda_j = A'_j(1)$ .

The service times of the class  $j$  packets are assumed to be i.i.d. and are characterized by the probability mass function  $s_j(m)$  and probability generating function  $S_j(z)$  ( $j = 1, 2$ ). We furthermore denote the mean service time of a class  $j$  packet by  $\mu_j = S'_j(1)$ . We define the load offered by class  $j$  packets as  $\rho_j \triangleq \lambda_j \mu_j$  ( $j = 1, 2$ ). The total load is then given by  $\rho \triangleq \rho_1 + \rho_2$ .

The system has one server that provides the transmission of packets. Class 1 packets are assumed to have non-preemptive priority over class 2 packets, and within one class the service discipline is FCFS. Due to the priority scheduling mechanism, it is as if class 1 packets are stored in front of class 2 packets in the queue. So, if there are any class 1 packets in the queue when the server becomes

idle, the one with the longest waiting time will be served next. If, on the other hand, no class 1 packets are present in the queue at that moment, the class 2 packet with the longest waiting time, if any, will be served next. Since the priority scheduling is non-preemptive, service of a packet will not be interrupted by newly arriving packets.

### 3 System Contents

To be able to analyze the packet delay, we will first analyse the system contents at the beginning of so-called start slots, i.e., slots at the beginning of which a packet (if available) can enter the server. Note that every slot during which the system is empty, is also a start slot. We denote the system contents of class  $j$  packets at the beginning of the  $l$ -th start slot by  $n_{j,l}$  ( $j = 1, 2$ ). Their joint pgf will be denoted by  $N_l(z_1, z_2)$ . Clearly, the set  $\{(n_{1,l}, n_{2,l})\}$  forms a Markov chain, since the arrival process is i.i.d. and only random variables during start slots are involved. If  $s^*$  indicates the service time of the packet that enters service at the beginning of start slot  $l$  (which is - by definition - regular slot  $k$ ) the following system equations can be established:

1. If  $n_{1,l} = n_{2,l} = 0$ :

$$n_{1,l+1} = a_{1,k} ; n_{2,l+1} = a_{2,k}, \quad (1)$$

i.e., the only packets present in the system at the beginning of start slot  $l+1$  are the packets that arrived during the previous slot, i.e., start slot  $l$ .

2. If  $n_{1,l} = 0$  and  $n_{2,l} > 0$ :

$$n_{1,l+1} = \sum_{i=0}^{s^*-1} a_{1,k+i} ; n_{2,l+1} = n_{2,l} + \sum_{i=0}^{s^*-1} a_{2,k+i} - 1, \quad (2)$$

i.e., the class 2 packet in service leaves the system just before start slot  $l+1$ .  $s^*$  is characterized by probability mass function  $s_2(m)$ , since a class 2 packet enters the server at the beginning of start slot  $l$ .

3. If  $n_{1,l} > 0$ :

$$n_{1,l+1} = n_{1,l} + \sum_{i=0}^{s^*-1} a_{1,k+i} - 1 ; n_{2,l+1} = n_{2,l} + \sum_{i=0}^{s^*-1} a_{2,k+i}, \quad (3)$$

i.e., the class 1 packet in service leaves the system just before start slot  $l+1$ .  $s^*$  is characterized by probability mass function  $s_1(m)$ , since a class 1 packet enters the server at the beginning of start slot  $l$ .

In the remainder, we define  $E[X\{Y\}]$  as  $E[X|Y]\text{Prob}[Y]$ . The system equations (1)-(3) can now be translated into relations between  $z$ -transforms. Exploiting

the statistical independence of the (set of) random variables  $s^*$ ,  $(n_{1,l}, n_{2,l})$  and  $(a_{1,k+i}, a_{2,k+i})$ ,  $i \geq 0$ , respectively, this leads to the following relation:

$$\begin{aligned} N_{l+1}(z_1, z_2) &\triangleq \mathbb{E} [z_1^{n_{1,l+1}} z_2^{n_{2,l+1}}] = \mathbb{E} [z_1^{n_{1,l+1}} z_2^{n_{2,l+1}} \{n_{1,l} = n_{2,l} = 0\}] \\ &\quad + \mathbb{E} [z_1^{n_{1,l+1}} z_2^{n_{2,l+1}} \{n_{1,l} = 0, n_{2,l} > 0\}] + \mathbb{E} [z_1^{n_{1,l+1}} z_2^{n_{2,l+1}} \{n_{1,l} > 0\}] \\ &= A(z_1, z_2) N_l(0, 0) + \frac{S_2(A(z_1, z_2))}{z_2} [N_l(0, z_2) - N_l(0, 0)] \\ &\quad + \frac{S_1(A(z_1, z_2))}{z_1} [N_l(z_1, z_2) - N_l(0, z_2)]. \end{aligned} \quad (4)$$

We assume that the system is stable (implying that the equilibrium condition requires that  $\rho < 1$ ) and as a result  $N_l(z_1, z_2)$  and  $N_{l+1}(z_1, z_2)$  converge both to a common steady-state limit denoted by  $N(z_1, z_2)$ . By taking the  $l \rightarrow \infty$  limit of equation (4), we obtain:

$$\begin{aligned} [z_1 - S_1(A(z_1, z_2))] N(z_1, z_2) &= z_1 \frac{z_2 A(z_1, z_2) - S_2(A(z_1, z_2))}{z_2} N(0, 0) \\ &\quad + \frac{z_1 S_2(A(z_1, z_2)) - z_2 S_1(A(z_1, z_2))}{z_2} N(0, z_2). \end{aligned} \quad (5)$$

It now remains for us to determine the unknown function  $N(0, z_2)$  and the unknown parameter  $N(0, 0)$ . This can be done in two steps. First, we notice that  $N(z_1, z_2)$  must be bounded for all values of  $z_1$  and  $z_2$  such that  $|z_1| \leq 1$  and  $|z_2| \leq 1$ . In particular, this should be true for  $z_1 = Y(z_2)$ , with  $Y(z_2) \triangleq S_1(A(Y(z_2), z_2))$  and  $|z_2| \leq 1$ , since it follows from Rouché's theorem that there is exactly one solution  $|Y(z_2)| \leq 1$  for all such  $z_2$ . Notice that  $Y(1)$  equals 1. The above implies that if we choose  $z_1 = Y(z_2)$  in equation (5), where  $|z_2| \leq 1$ , the left hand side of this equation vanishes. The same must then be true for the right hand side, yielding

$$N(0, z_2) = N(0, 0) \frac{z_2 A(Y(z_2), z_2) - S_2(A(Y(z_2), z_2))}{z_2 - S_2(A(Y(z_2), z_2))}. \quad (6)$$

Finally, in order to find an expression for  $N(0, 0)$ , we put  $z_1 = z_2 = 1$  and use de l'Hospital's rule in equation (5). Therefore, we need the first derivative of  $Y(z)$  for  $z = 1$  and this is given by

$$Y'(1) = \mu_1(\lambda_1 Y'(1) + \lambda_2) = \frac{\lambda_2 \mu_1}{1 - \rho_1}. \quad (7)$$

We then obtain  $N(0, 0)$ :

$$N(0, 0) = \frac{1 - \rho}{1 - \rho + \lambda_1 + \lambda_2}. \quad (8)$$

A fully determined expression for  $N(z_1, z_2)$  can now be derived by combining equations (5) and (6):

$$\begin{aligned}
 N(z_1, z_2) = N(0, 0) & \left[ \frac{z_1(z_2 A(z_1, z_2) - S_2(A(z_1, z_2)))}{(z_1 - S_1(A(z_1, z_2)))(z_2 - S_2(A(Y(z_2), z_2)))} \right. \\
 & + \frac{S_2(A(Y(z_2), z_2))(S_1(A(z_1, z_2)) - z_1 A(z_1, z_2))}{(z_1 - S_1(A(z_1, z_2)))(z_2 - S_2(A(Y(z_2), z_2)))} \\
 & \left. + \frac{A(Y(z_2), z_2)(z_1 S_2(A(z_1, z_2)) - z_2 S_1(A(z_1, z_2)))}{(z_1 - S_1(A(z_1, z_2)))(z_2 - S_2(A(Y(z_2), z_2)))} \right], \quad (9)
 \end{aligned}$$

with  $N(0, 0)$  given by equation (8) and  $Y(z)$  implicitly defined by  $Y(z) = S_1(A(Y(z), z))$ .

## 4 Packet Delay

The packet delay is defined as the total time period a tagged packet spends in the system, i.e., the number of slots between the end of the packet's arrival slot and the end of its departure slot. We denote the delay of a tagged class  $j$  packet by  $d_j$  and its pgf by  $D_j(z)$  ( $j = 1, 2$ ). Before deriving expressions for  $D_1(z)$  and  $D_2(z)$ , we first define some stochastic variables we will frequently use in this section. We denote the arrival slot of the tagged packet by slot  $k$ . If slot  $k$  is a start slot, it is assumed to be start slot  $l$ . If slot  $k$  is not a start slot on the other hand, the last start slot preceeding slot  $k$  is assumed to be start slot  $l$ . We denote the number of class  $j$  packets that arrive during slot  $k$ , but which are served before the tagged packet by  $f_j$  ( $j = 1, 2$ ). We denote the service time of the tagged class  $j$  packet by  $s_j^*$  ( $j = 1, 2$ ). We finally denote the service time and the elapsed service time of the packet in service (if any) during the arrival slot of the tagged packet by  $s^*$  and  $s^+$  respectively.

### 4.1 Delay of Class 1 Packets

We tag a class 1 packet. There are 3 possibilities when the tagged packet arrives:

1. The server is idle during slot  $k$ , yielding

$$d_1 = \sum_{m=1}^{f_1} s_{1,m}^{(k)} + s_1^*, \quad (10)$$

with the  $s_{1,m}^{(k)}$ 's the service times of the class 1 packets that arrived during slot  $k$ , but that are served before the tagged class 1 packet.

2. A class 2 packet is in service during slot  $k$  (implying that  $n_{1,l} = 0$ ,  $n_{2,l} > 0$ ), yielding

$$d_1 = (s^* - s^+ - 1) + \sum_{i=1}^{s^+} \sum_{m=1}^{a_{1,k-i}} s_{1,m}^{(k-i)} + \sum_{m=1}^{f_1} s_{1,m}^{(k)} + s_1^*, \quad (11)$$

with the  $s_{1,m}^{(k-i)}$ 's ( $0 \leq i \leq s^+$ ) the service times of the class 1 packets that arrived during slot  $k-i$ . The residual service time of the packet in service during slot  $k$  contributes in the first term, the service times of the class 1 packets in the system at the beginning of slot  $k$  contribute in the second term, the service times of the class 1 packets arrived during slot  $k$ , but served before the tagged class 1 packet contribute in the third term, and finally the service time of the tagged class 1 packet itself contributes in the last term.

3. A class 1 packet is in service during slot  $k$  (i.e.,  $n_{1,l} > 0$ ), yielding

$$d_1 = (s^* - s^+ - 1) + \sum_{m=1}^{n_{1,l}-1} \tilde{s}_{1,m} + \sum_{i=1}^{s^+} \sum_{m=1}^{a_{1,k-i}} s_{1,m}^{(k-i)} + \sum_{m=1}^{f_1} s_{1,m}^{(k)} + s_1^*. \quad (12)$$

The difference with the previous situation is that there may be multiple high priority packets in the buffer (apart from the one in service) at the beginning of slot  $l$ , which will contribute to the tagged packet's delay. If we denote by  $\tilde{s}_{1,m}$  the service times of the class 1 packets already in the queue at the beginning of the ongoing service (thus without the packet in service during slot  $k$ ), then this condition is quantified by the second term in the right-hand side of the above expression.

Again, equations (10)-(12) can be  $z$ -transformed. Taking the sum then eventually leads to an expression for  $D_1(z)$ :

$$\begin{aligned} D_1(z) &\triangleq E[z^{d_1}] = E[z^{d_1} \{\text{no service}\}] + E[z^{d_1} \{\text{service class 2 packet}\}] \\ &\quad + E[z^{d_1} \{\text{service class 1 packet}\}] \\ &= F_1(S_1(z))S_1(z) \left\{ 1 - \rho + \rho_2 \frac{S_2^* \left( \frac{A_1(S_1(z))}{z} \right), z}{z} \right. \\ &\quad \left. + \rho_1 \frac{S_1^* \left( \frac{A_1(S_1(z))}{z} \right), z}{z} \frac{N(S_1(z), 1) - N(0, 1)}{(1 - N(0, 1))S_1(z)} \right\}, \end{aligned} \quad (13)$$

with  $F_1(z) \triangleq E[z^{f_1}]$ ,  $S_2^*(x, z) \triangleq E[x^{s^+} z^{s^*} | n_{1,l} = 0, n_{2,l} > 0]$  and  $S_1^*(x, z) \triangleq E[x^{s^+} z^{s^*} | n_{1,l} > 0]$ . The random variable  $f_1$  can be shown to have the following pgf (see e.g. [2]):

$$F_1(z) = \frac{A_1(z) - 1}{\lambda_1(z - 1)}. \quad (14)$$

If a class  $j$  packet is in service during slot  $k$ ,  $s^*$  is characterized by the probability mass function  $s_j(m)$  ( $j = 1, 2$ ). Notice that the distributions of  $s^*$  and  $s^+$  are correlated, since  $s^+$  is the elapsed part of the service time  $s^*$  at the beginning of

slot  $k$ . Considering these observations, one can derive the following expression for  $S_j^*(x, z)$ :

$$S_j^*(x, z) = \frac{S_j(xz) - S_j(z)}{\mu_j(x - 1)}, \quad (15)$$

with  $j = 1, 2$ . Substitution of (9), (14) and (15) into equation (13) finally leads to a closed-form version of  $D_1(z)$ :

$$D_1(z) = \frac{(1 - \rho)(z - 1) + \lambda_2(S_2(z) - 1)}{\lambda_1(S_1(z) - 1)} \frac{S_1(z)(A_1(S_1(z)) - 1)}{z - A_1(S_1(z))}. \quad (16)$$

## 4.2 Delay of Class 2 Packets

Because of the priority discipline, an expression for  $d_2$  will be a bit more involved. We now tag a class 2 packet that enters the buffer during slot  $k$ . Let us refer to the packets in the system at the end of slot  $k$ , but that have to be served before the tagged packet as the “primary packets”. So, basically, the tagged class 2 packet can enter the server, when all primary packets and all class 1 packets that arrived after slot  $k$  are transmitted. In order to analyse the delay of the tagged class 2 packet, the number of class 1 packets and class 2 packets that are served between the arrival slot of the tagged class 2 packet and its departure slot is important, not the precise order in which they are served. Therefore, in order to facilitate the analysis, we will consider an equivalent virtual system with an altered service discipline. We assume that from slot  $k$  on, the order of service for class 1 packets (those in the queue at the end of slot  $k$  and newly arriving ones) is LCFS instead of FCFS in the equivalent system (the transmission of class 2 packets remains FCFS). So, a primary packet can enter the server, when the system becomes free (for the first time) of class 1 packets that arrived during and after the service time of the primary packet that predeceased it according to the new service discipline. Let  $v_{1,m}^{(i)}$  denote the length of the time period during which the server is occupied by the  $m$ -th class 1 packet that arrives during slot  $i$  and its class 1 “successors”, i.e., the time period starting at the beginning of the service of that packet and terminating when the system becomes free (for the first time) of class 1 packets which arrived during and after its service time. Analogously, let  $v_{2,m}^{(i)}$  denote the length of the time period during which the server is occupied by the  $m$ -th class 2 packet that arrives during slot  $i$  and its class 1 “successors”. The  $v_{j,m}^{(i)}$ ’s ( $j = 1, 2$ ) are called sub-busy periods, caused by the  $m$ -th class  $j$  packet that arrived during slot  $i$ .

When the tagged class 2 packet arrives, there are 3 possibilities:

1. The server is idle during slot  $k$ , yielding

$$d_2 = \sum_{j=1}^2 \sum_{m=1}^{f_j} v_{j,m}^{(k)} + s_2^*, \quad (17)$$

i.e.,  $f_1$  class 1 primary packets and  $f_2$  class 2 primary packets that arrived during slot  $k$  and their class 1 successors have to be served before the tagged class 2 packet.

2. A class 2 packet is in service during slot  $k$ , yielding

$$d_2 = (s^* - s^+ - 1) + \sum_{i=1}^{s^* - s^+ - 1} \sum_{m=1}^{a_{1,k+i}} v_{1,m}^{(k+i)} + \sum_{j=1}^2 \sum_{m=1}^{f_j} v_{j,m}^{(k)} \quad (18)$$

$$+ \sum_{j=1}^2 \sum_{i=1}^{s^+} \sum_{m=1}^{a_{j,k-i}} v_{j,m}^{(k-i)} + \sum_{m=1}^{n_{2,l}-1} \tilde{v}_{2,m} + s_2^*,$$

with the  $\tilde{v}_{2,m}$ 's the sub-busy periods, caused by the  $m$ -th class 2 packet already in the queue at the beginning of start slot  $l$ . The residual service time of the packet in service during slot  $k$  contributes in the first term, the sub-busy periods of the class 1 packets arriving during the residual service time contribute in the second term, the sub-busy periods of the class 1 and class 2 packets arriving during slot  $k$ , but that have to be served before the tagged class 2 packet contribute in the third term, the sub-busy periods of the class 1 and class 2 packets that arrived during the elapsed service time contributes in the fourth term, the sub-busy period of the class 2 packets already in the queue at the beginning of start slot  $l$  contributes in the fifth term and finally the service time of the tagged class 2 packet itself contributes in the last term.

3. A class 1 packet is in service during slot  $k$ , yielding

$$d_2 = (s^* - s^+ - 1) + \sum_{i=1}^{s^* - s^+ - 1} \sum_{m=1}^{a_{1,k+i}} v_{1,m}^{(k+i)} + \sum_{j=1}^2 \sum_{m=1}^{f_j} v_{j,m}^{(k)} \quad (19)$$

$$+ \sum_{j=1}^2 \sum_{i=1}^{s^+} \sum_{m=1}^{a_{j,k-i}} v_{j,m}^{(k-i)} + \sum_{m=1}^{n_{1,l}-1} \tilde{v}_{1,m} + \sum_{m=1}^{n_{2,l}} \tilde{v}_{2,m} + s_2^*,$$

with the  $\tilde{v}_{j,m}$ 's ( $j = 1, 2$ ) the sub-busy periods, caused by the  $m$ -th class  $j$  packet already in the queue at the beginning of start slot  $l$ . The expression is virtually the same as in the previous case, with an additional term that takes into account the sub-busy periods of the class 1 packets already in the system when the transmission of the class 1 packet currently in the server started (i.e., at the beginning of slot  $l$ ).

Due to the initial assumptions and since the length of different sub-busy periods only depends on the number of class 1 packet arrivals during different slots and the service times of the corresponding primary packets, the sub-busy periods associated with the primary packets of class 1 and class 2 form a set of i.i.d. random variables and their pgf will be presented by  $V_1(z)$  and  $V_2(z)$  respectively. Notice that  $f_1$  and  $f_2$  are correlated; in section 2 it was explained that  $a_{1,k}$  and  $a_{2,k}$  may be correlated as well. Once again, applying a  $z$ -transform technique to

equations (17)-(19) and taking into account the previous remarks, we can derive an expression for  $D_2(z)$ :

$$\begin{aligned}
 D_2(z) &\triangleq E[z^{d_2}] = E[z^{d_2}\{\text{no service}\}] + E[z^{d_2}\{\text{service class 2 packet}\}] \\
 &\quad + E[z^{d_2}\{\text{service class 1 packet}\}] \\
 &= F(V_1(z), V_2(z))S_2(z) \left\{ 1 - \rho + \rho_2 \frac{S_2^* \left( \frac{A(V_1(z), V_2(z))}{zA_1(V_1(z))}, zA_1(V_1(z)) \right)}{zA_1(V_1(z))} \right. \\
 &\quad \left. \frac{N(0, V_2(z)) - N(0, 0)}{(N(0, 1) - N(0, 0))V_2(z)} + \rho_1 \frac{S_1^* \left( \frac{A(V_1(z), V_2(z))}{zA_1(V_1(z))}, zA_1(V_1(z)) \right)}{zA_1(V_1(z))} \right. \\
 &\quad \left. \frac{N(V_1(z), V_2(z)) - N(0, V_2(z))}{(1 - N(0, 1))V_1(z)} \right\}, \tag{20}
 \end{aligned}$$

with  $F(z_1, z_2) \triangleq E[z_1^{f_1} z_2^{f_2}]$ ,  $S_2^*(x, z) \triangleq E[x^{s^+} z^{s^*} | n_{1,l} = 0, n_{2,l} > 0]$  and  $S_1^*(x, z) \triangleq E[x^{s^+} z^{s^*} | n_{1,l} > 0]$ . The random variables  $f_1$  and  $f_2$  can be shown to have the following joint pgf (extension of a technique used in e.g. [2]):

$$F(z_1, z_2) = \frac{A(z_1, z_2) - A_1(z_1)}{\lambda_2(z_2 - 1)}. \tag{21}$$

The  $S_j^*(x, z)$ 's ( $j = 1, 2$ ) are again given by equation (15). Finally, we have to find expressions for  $V_1(z)$  and  $V_2(z)$ . These pgfs satisfy the following relations:

$$V_j(z) = S_j(zA_1(V_1(z))), \tag{22}$$

with ( $j = 1, 2$ ). This can be understood as follows: when the  $m$ -th class  $j$  packet that arrived during slot  $i$  enters service,  $v_{j,m}^{(i)}$  consists of two parts: the service time of that packet itself, and the service times of the class 1 packets that arrive during its service time and of their class 1 successors. This leads to equation (22). Equation (20) together with equations (21) and (15) leads to a fully determined version for  $D_2(z)$ :

$$D_2(z) = \frac{1 - \rho}{\lambda_2} \frac{S_2(z)(A(V_1(z), V_2(z)) - A_1(V_1(z)))}{zA_1(V_1(z)) - A(V_1(z), V_2(z))} \frac{1 - zA_1(V_1(z))}{1 - V_2(z)}. \tag{23}$$

## 5 Calculation of Moments

The functions  $Y(z)$ ,  $V_1(z)$  and  $V_2(z)$  can only be explicitly found in case of some simple arrival processes. Their derivatives for  $z = 1$ , necessary to calculate the moments of the system contents and the cell delay, on the contrary, can be calculated in closed-form. For example,  $Y'(1)$  is given by equation (7) and the first derivatives of  $V_j(z)$  for  $z = 1$  are given by

$$V_j'(1) = \frac{\mu_j}{1 - \rho_1},$$



with  $(j = 1, 2)$ . Let us define  $\lambda_{ij}$  and  $\mu_{jj}$  as

$$\lambda_{ij} \triangleq \left. \frac{\partial^2 A(z_1, z_2)}{\partial z_i \partial z_j} \right|_{z_1=z_2=1} ; \quad \mu_{jj} \triangleq \left. \frac{d^2 S_j(z)}{dz^2} \right|_{z=1},$$

with  $i, j = 1, 2$ . Now we can calculate the mean values of the packet delay of both classes by taking the first derivatives of the respective pgfs for  $z = 1$ . We find

$$E[d_1] = \mu_1 + \frac{1}{2} \frac{\mu_1 \lambda_{11}}{\lambda_1(1 - \rho_1)} + \frac{1}{2} \frac{\lambda_1 \mu_{11} + \lambda_2 \mu_{22}}{1 - \rho_1},$$

for the mean value of the packet delay of a class 1 packet and

$$E[d_2] = \mu_2 + \frac{1}{2} \frac{\mu_1^2 \lambda_{11}}{(1 - \rho)(1 - \rho_1)} + \frac{1}{2} \frac{2\mu_1 \lambda_{12} + \mu_2 \lambda_{22}}{\lambda_2(1 - \rho)} + \frac{1}{2} \frac{\lambda_1 \mu_{11} + \lambda_2 \mu_{22}}{(1 - \rho)(1 - \rho_1)},$$

for the mean value of the packet delay of a class 2 packet. In a similar way, expressions for the variance (or higher order moments) can be calculated as well by taking the appropriate derivatives of the respective generating functions.

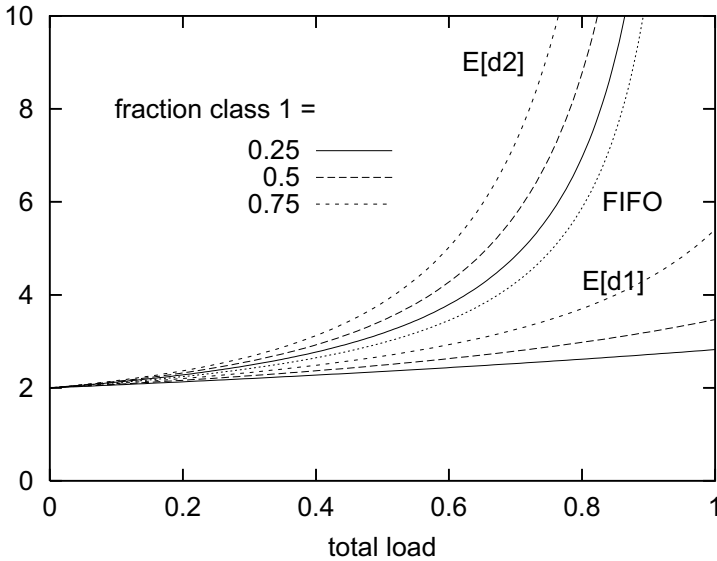
## 6 Numerical Examples

In this section, we present some numerical examples. We assume the traffic of the two classes to be arriving according to a two-dimensional binomial process. Its two-dimensional pgf is given by:

$$A(z_1, z_2) = \left(1 - \frac{\lambda_1}{N}(1 - z_1) - \frac{\lambda_2}{N}(1 - z_2)\right)^N. \quad (24)$$

The arrival rate of class  $j$  traffic is thus given by  $\lambda_j$  ( $j = 1, 2$ ). This arrival process occurs for instance at an output queue of a  $N \times N$  switch fed by a Bernoulli process at the inlets (see [16]). Notice also that if  $N \rightarrow \infty$ , the arrival process becomes a superposition of two independent Poisson streams. In the remainder of this section, we assume that  $N = 16$ . We will furthermore assume deterministic service times for both classes.

In Fig. 1., the mean value of the packet delay of class 1 packets and class 2 packets is shown as a function of the total load  $\rho$ , when  $\mu_1 = \mu_2 = 2$ . The fraction of class 1 arrivals is 0.25, 0.5 and 0.75 respectively of the total number of arrivals. In order to compare with FIFO scheduling, we have also shown the mean value of the packet delay in that case. Since, in this example, the service times of the class 1 and class 2 packets are equal, the packet delay is then of course the same for class 1 and class 2 packets, and can thus be calculated as if there is only one class of packets arriving according to an arrival process with pgf  $A(z, z)$ . This situation has already been analyzed, e.g., in [2]. One can observe the influence of priority scheduling: mean delay of class 1 packets reduces significantly. The price to pay is of course a larger mean delay for class 2 packets. If this kind of



**Fig. 1.** Mean packet delay when the fraction of class 1 arrivals equals 0.25, 0.5 and 0.75

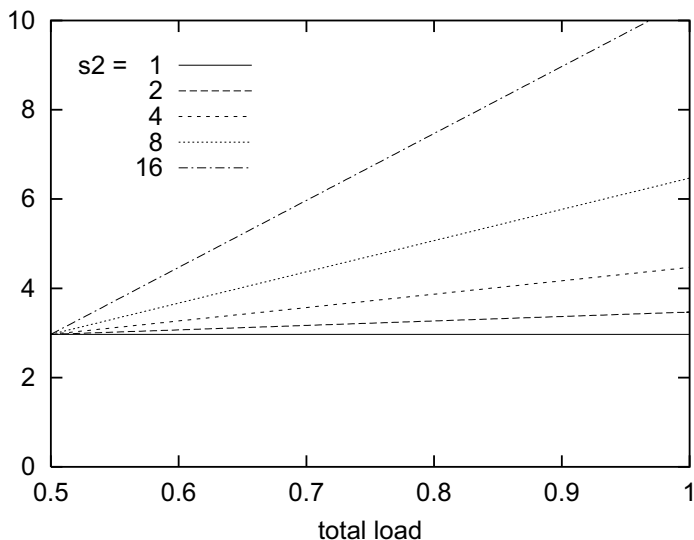
traffic is not delay-sensitive, as assumed, this is not a too big a problem. Also, the smaller the fraction of high priority packets in the overall traffic mix, the lower the mean packet delay of both classes will be.

Fig. 2. shows the mean value of the packet delay of class 1 packets as a function of the total load, when  $\lambda_1 = 0.25$ ,  $\mu_1 = 2$  and  $\mu_2 = 1, 2, 4, 8, 16$ . This figure shows the influence of the non-preemptive priority scheduling. When the service time of a class 2 packet is assumed to be deterministically equal to 1 slot, i.e.,  $\mu_2 = 1$ , the preemptive priority scheduling has the same effect as the non-preemptive priority scheduling. If  $\mu_2 > 1$ , the non-preemptive priority has worse performance than the preemptive priority scheduling in terms of the mean packet delay for class 1 packets. Furthermore, for a given value of the low priority packet length, the mean high priority packet delay increases proportional to the total load  $\rho$ .

## 7 Conclusions

In this paper, we analyzed the packet delay in a queueing system with non-preemptive HOL priority scheduling. A generating-functions-approach was adopted, which led to closed-form expressions of performance measures, such as mean of packet delay of both classes, that are easy to evaluate. The model included possible correlation between the number of arrivals of the two classes during a slot and general service times for packets of both classes. Therefore, the

results can be applied to analyse the performance of traffic streams in an environment with delay and loss sensitive traffic, such as an integrated voice/data IP network.



**Fig. 2.** Mean packet delay of class 1 packets when the service time of class 2 packets equals 1, 2, 4, 8, 16

## References

1. J.J. Bae and T. Suda, *Survey of traffic control schemes and protocols in ATM networks*, Proceedings of the IEEE 79(2), pp. 170-189, 1991.
2. H. Bruneel and B.G. Kim, *Discrete-time models for communication systems including ATM*, Kluwer Academic Publishers, Boston, 1993.
3. N.K. Jaiswal, *Priority queues*, Academic Press, New York, 1968.
4. A. Khamisy and M. Sidi, *Discrete-time priority queues with two-state Markov modulated arrivals*, Stochastic Models 8(2), pp. 337-357, 1992.
5. K. Laevens and H. Bruneel, *Discrete-time multiserver queues with priorities*, Performance Evaluation 33(4), pp. 249-275, 1998.
6. K. Liu, D.W. Petr and V.S. Frost, *Design and analysis of a bandwidth management framework for ATM-based broadband ISDN*, IEEE Communications Magazine, pp. 138-145, 1997.
7. I. Rubin and Z. Tsai, *Message delay analysis of multiclass priority TDMA, FDMA, and discrete-time queueing systems*, IEEE Transactions on Information Theory 35(3), pp. 637-647, 1989.
8. D.A. Stanford, *Interdeparture-time distributions in the non-preemptive priority  $\sum M_i/G_i/1$  queue*, Performance Evaluation 12, pp. 43-60, 1991.

9. A. Sugahara, T. Takine, Y. Takahashi and T. Hasegawa, *Analysis of a nonpreemptive priority queue with SPP arrivals of high class*, Performance Evaluation 21, pp. 215-238, 1995.
10. L. Takacs, *Priority queues*, Operations Research 12, pp. 63-74, 1964.
11. H. Takagi, *Queueing analysis A foundation of Performance Evaluation Volume 1: Vacation and priority systems*, North-Holland, 1991.
12. T. Takine, Y. Matsumoto, T. Suda and T. Hasegawa, *Mean waiting times in non-preemptive priority queues with Markovian arrival and i.i.d. service processes*, Performance Evaluation 20, pp. 131-149, 1994.
13. T. Takine, B. Sengupta and T. Hasegawa, *An analysis of a discrete-time queue for broadband ISDN with priorities among traffic classes*, IEEE Transactions on Communications 42 (2-4), pp. 1837-1845, 1994.
14. T. Takine, *A nonpreemptive priority MAP/G/1 queue with two classes of customers*, Performance Evaluation 20, pp. 266-290, 1996.
15. P. Van Mieghem, B. Steyaert and G.H. Petit, *Performance of cell loss priority management schemes in a single server queue*, International Journal of Communication Systems 10, pp. 161-180, 1997.
16. J. Walraevens and H. Bruneel, *HOL priority in an ATM output queueing switch*, Proceedings of the seventh IFIP workshop on performance modelling and evaluation of ATM/IP networks, Antwerp, 28-30 june, 1999.

# New Results on the Numerical Stability of the Stochastic Fluid Flow Model Analysis

Markus Fiedler<sup>1</sup> and Holger Voos<sup>2</sup>

<sup>1</sup> University of Karlskrona/Ronneby

Dep. of Telecommunications and Signal Processing (ITS)

S-37179 Karlskrona, Sweden

phone: +46-708-537339, fax: +46-455-385667 [markus.fiedler@its.hk-r.se](mailto:markus.fiedler@its.hk-r.se)

<sup>2</sup> University of Kaiserslautern, Institute of Process Automation

PO-Box 3049, D-67653 Kaiserslautern, Germany

phone: +49-631-205 4457, fax: +49-631-205 4462, [voos@eit.uni-kl.de](mailto:voos@eit.uni-kl.de)

**Abstract.** The stochastic fluid flow model (SFF) is one of the leading models in performance evaluation for tele- and datacommunication systems, especially in fast packet-switching networks and ATM. However, the numerical analysis of the SFF is widely considered to be unstable. In this paper, some investigations and results are presented concerning the numerical stability of the SFF analysis also for large systems with finite buffer. We identify the main source of the numerical problems and give hints how to circumvent them. The usefulness of different solution methods are compared and the most robust methods for systems with large numbers of sources and large buffer sizes are identified.

## 1 Introduction

One possible model for performance evaluation in communications systems is the so called stochastic fluid flow model (SFF). It is widely used when dealing with fast packet-switched and ATM networks but also allows performance analysis in any kind of tele- and datacommunications system including mobile communications. First pioneering work considering the SFF can be found in [8]. In [1], an elegant treatment of a finite number of homogeneous on-off fluid sources whose streams were concentrated by a multiplexer with infinite buffer is presented. In 1984, Kosten presented an expansion of that model to heterogeneous traffic [9], and in 1988, Tucker published the formal way of how to treat finite multiplexer buffers [17]. The main advantage of the SFF is the computational complexity which is independent of the buffer size [12].

However, the analysis of the SFF is widely considered to be numerically unstable. Indeed, the solution contains components that are exponentially increasing with the buffer content in the case of limited buffer size [8], [1], [17]. For that reason and the fact that no closed-form solutions exist, most work in the 1990's deals with buffers of infinite size or approximations of the solution based upon that assumption. In 1991, Nagarajan et al [13] reported results for 130 voice sources described in [7], which clearly differ from their simulation results.

In 1995, Yang et al [18] described results for 25 homogeneous sources but were unable to obtain results for large-size problems due to numerical instability of their solution method. In spite of these difficulties, very few material exist that discusses these numerical problems in detail.

Therefore, this paper closes that gap between theory and numerical practice for the classical *spectral method* that is based on computing an eigensystem (*spectral decomposition* and on determining the coefficients associated to the *spectral components*. We are going to demonstrate how stable the spectral method actually might perform even for large systems with finite buffers if some quite simple rules are taken care of. In section 2, we describe the model and the notation that is used throughout this paper. Section 3 deals with the main steps of the stochastic fluid flow analysis. Section 4 introduces the numerical methods to solve the system of linear equations, which is necessary to adapt the solution of the system of differential equations to the boundary conditions. Furthermore, some special implementation issues are proposed. Section 5 presents numerical results and discusses the feasibility of different numerical methods for systems with homogeneous and heterogeneous sources. Section 6 summarizes the main results and gives an outlook on future work.

## 2 The Stochastic Fluid Flow Model

The SFF under investigation is the classical model which can be found in many publications. We assume  $N$  on-off fluid sources that alternate randomly between an on state with peak cell rate  $h$  and an off state, both of exponentially distributed duration. Thus, a discrete-valued rate process  $S$  is formed with  $n_S$  states  $s_i$  and flow intensities  $R \in \{r_i\}, i \in \{0, \dots, n_S - 1\}$ . In addition, we assume a fluid buffer of finite size  $K$  and an outlet (= server) with capacity  $C$ . The dynamic of the rate process  $S$  is described by an irreducible Markov chain, represented by the infinitesimal generator matrix  $\mathbf{M}$ , which also determines the state probabilities  $\pi_i$ . Each state has a drift value  $d_i = r_i - C$ , depending on which the states are classified in

- over-load states  $\mathcal{S}^o = \{i \mid d_i > 0\}$ ;
- equilibrium states  $\mathcal{S}^e = \{i \mid d_i = 0\}$ ;
- under-load states  $\mathcal{S}^u = \{i \mid d_i < 0\}$ .

These values are collected in the so called drift matrix  $\mathbf{D} = \text{diag}[d_i]$ . The matrix  $\mathbf{R} = \text{diag}[r_i]$  is called rate matrix. We denote the aggregate state space with  $\mathcal{S} = \mathcal{S}^o \cup \mathcal{S}^e \cup \mathcal{S}^u$ . The  $n$  on-off sources might be homogeneous, i. e. all have the same parameters, or heterogeneous. In the latter case, we form  $n_G$  groups, each containing homogeneous sources.

## 3 The Fluid Flow Analysis

The following represents a summary of the fluid flow analysis which deals with the most essential points in the numerical context. Further information might be obtained from [1], [10], [15], [17].

Let  $X$  be the buffer content of the fluid flow buffer with  $0 \leq X < K$ . Its stationary distribution function  $\mathbf{F}(x)$  with elements  $F_i(x) = \Pr\{X \leq x \wedge \text{state} = i\}$  is governed by the system of differential equations

$$\mathbf{D} \cdot \frac{d}{dx} \mathbf{F}(x) = \mathbf{M} \cdot \mathbf{F}(x). \quad (1)$$

From (1), an eigenvalue-eigenvector problem

$$z_q \mathbf{D} \cdot \boldsymbol{\varphi}_q = \mathbf{M} \cdot \boldsymbol{\varphi}_q \quad (2)$$

with  $q \in \{0, \dots, n_S - 1\}$  is obtained. The eigenvalues  $z_q$  and eigenvectors  $\boldsymbol{\varphi}_q$  appear in the so called spectral components in the solution of (1),

$$\mathbf{F}(x) = \sum_{q \in \mathcal{S}} a_q(K) \boldsymbol{\varphi}_q \exp(z_q x). \quad (3)$$

For homogeneous on-off sources, eigenvalues and eigenvectors are given in closed form [1]. We normalize the eigenvectors in a way that the sum of all elements equals one. If the sources are heterogeneous, the set of eigenvalues  $\{z_q\}$  has to be determined numerically from the inverse eigenvalue problem

$$\gamma_q(z_q) \boldsymbol{\varphi}_q = \left( \mathbf{R} - \frac{1}{z_q} \mathbf{M} \right) \boldsymbol{\varphi}_q \quad \text{with} \quad \gamma_q(z_q) = \sum_{j=1}^{n_G} \gamma_q^{(j)}(z_q) = C. \quad (4)$$

For groups of on-off sources, the functions  $\gamma_q^{(j)}(z_q)$  are also given in closed form and depend on the number of sources that are on in state  $q$  and belong to group  $j$ . Once  $z_q$  is determined, the corresponding eigenvector is a composition of parts that are determined for homogeneous groups. This composition is done using Kronecker algebra. More details might be found in [15], [10].

The coefficients  $a_q(K)$  in (3) are necessary to adjust the solution to the boundary conditions, which depend on the buffer size  $K$ :

$$F_i(K) = \pi_i, \quad i \in \mathcal{S}^u; \quad F_i(0) = 0, \quad i \in \mathcal{S}^o. \quad (5)$$

Insertion of (5) into (3) leads to a system of linear equations that has to be solved in order to obtain the coefficients:

$$\begin{aligned} \sum_{q \in \mathcal{S}^u \cup \mathcal{S}^o} a_q(K) \varphi_{qi} \exp(z_q K) &= \pi_i; \quad i \in \mathcal{S}^u \\ \sum_{q \in \mathcal{S}^u \cup \mathcal{S}^o} a_q(K) \varphi_{qi} &= 0; \quad i \in \mathcal{S}^o \end{aligned} \quad (6)$$

Originally, cases  $d_i = 0$  had to be excluded by choosing an appropriate value of  $C$  [1]. However, we are able to deal with those equilibrium states in the same formal way as with over- or under-load states. Numerical investigations show

that if the drift in a non-equilibrium state becomes arbitrarily small, the corresponding coefficient also escapes in the limit while the corresponding eigenvector approaches a unity vector in  $q$ -direction:

$$\lim_{d_q \rightarrow 0} a_q(K) = 0, \quad \lim_{d_q \rightarrow 0} \varphi_q = \mathbf{e}_q \quad (7)$$

This means that in the limit  $d_q \rightarrow 0$ , there is no coupling between equation  $q$  and the other equations. Thus, we are free to ignore states with vanishing drift during the solution procedure.

### 3.1 Finite Buffer Case

The mostly ill-conditioned system of linear equations (6) of size  $(\dim \mathcal{S}^u + \dim \mathcal{S}^o) \leq n_S$  has to be solved numerically. Different methods to do this will be presented and evaluated in sections 4 and 5. As soon as the coefficients are determined, the probability that the buffer is full in state  $i$  is found to be

$$u_i(K) = \pi_i - \lim_{b \rightarrow K} F_i(b) = \pi_i - \sum_{q \in \mathcal{S}} a_q(K) \varphi_q \exp(z_q K), \quad (8)$$

see [17]. With this, the loss probability can be expressed as

$$P_L(K) = \frac{\sum_{i \in \mathcal{S}^o} u_i(K) d_i}{\mathbf{E}[R]}. \quad (9)$$

### 3.2 Buffer-Less Fluid Flow Model

The so called buffer-less fluid flow model is obtained by passing the buffer size to the limit  $K \rightarrow 0$ . This case may be assumed if the buffer is much smaller than the mean burst length and thus (almost) loses its influence on burst level. In this case, the fluid flow analysis described before doesn't need to be carried out anymore which allows the computation of systems with very large state spaces. Loss happens as soon as positive drift occurs, so that the probabilities  $u_i$  in the loss probability formula (9) can be replaced by the corresponding state probabilities  $\pi_i$ :

$$P_L(0) = \frac{\sum_{i \in \mathcal{S}^o} \pi_i d_i}{\mathbf{E}[R]}. \quad (10)$$

## 4 Numerical Methods

In this section we present a selection of methods that have been used to solve the system of linear equations (6) which is mostly ill-conditioned, not sparse, not symmetrical and not positive definite. In addition, the systems are large (depending on the size of the state space  $\mathcal{S}$ ) and associated with numerical input errors that stem from the numerical determination of the eigensystem. Our aim is to check how the following methods work in spite of these difficulties. Since the classification of the methods is not unique in the literature, we refer to the classification given by Stewart [16].



## 4.1 Direct Methods

With *direct methods*, we describe “numerical methods that compute solutions of mathematical problems in a fixed number of operations” [16]. We focus on methods that transform a system of linear equations  $\mathbf{A} \cdot \mathbf{x} = \mathbf{b}$  into a system  $\mathbf{A}' \cdot \mathbf{x} = \mathbf{b}'$  during a so-called reduction phase. Herein,  $\mathbf{A}'$  represents an upper triangular matrix. Then, the components of the solution vector  $\mathbf{x}$  are found by back substitution. See [5], [16], [4] for details on the following methods:

1. *Gaussian elimination*. The reduction is achieved by a special succession of divisions and subtractions of rows. Assume that in step  $i$  the first  $i - 1$  rows have already been treated. The elements of row  $i$  are obtained by

$$a_{kl}^{(i)} = a_{kl}^{(i-1)} - \frac{a_{ki}^{(i-1)}}{a_{ii}^{(i-1)}} a_{il}^{(i-1)} \quad k > i, l = 1 \dots n. \quad (11)$$

The elements  $a_{ii}^{(i-1)}$  are called pivots.

2. *Gaussian elimination with partial pivoting*. From a numerical point of view, the elements on the diagonal might not represent optimal pivots. It can be shown that the absolute value of the pivot should be as large as possible. Hence, if a pivot with larger absolute value can be obtained from another row in the same column, then the corresponding rows are interchanged.
3. *Gaussian elimination with full pivoting*. Here, an optimal pivot is searched in rows and columns. However, interchanging of columns leads to interchanged elements in the solution vector (a system  $\mathbf{A}' \cdot \mathbf{x}' = \mathbf{b}'$  appears), which have to be re-changed after back substitution.
4. *Givens rotations*. The reduction is performed by multiplying the system of equations with elementary rotation matrices, hereby cancelling out the elements in the lower triangular part. In [16], this method can be found among *projection methods*.
5. *Householder transformation*. This method also uses transformation matrices for the reduction phase. For details, see [11].

The methods 3 to 5 are stable with regard to the algorithm error.

## 4.2 Iteration Methods

*Iteration methods* try to approximate the solution by carrying out iterations, consisting of a couple of operations, as long as the approximation  $\mathbf{x}^*$  has not yet converged to a desired extent. Starting from  $\mathbf{x}^{*(0)} = \mathbf{0}$ , the result of an iteration serves as basis for the next iteration, i. e.  $\mathbf{x}^{*(i+1)} = f(\mathbf{x}^{*(i)})$ . To get a positive definite matrix instead of  $\mathbf{A}$  that guarantees convergence [5], we first apply a so called *Gaussian transformation* by multiplying our system of equations with the transposed matrix  $\mathbf{A}^T$  from the left side, i.e. we treat the system  $\mathbf{A}^T \cdot \mathbf{A} \cdot \mathbf{x} = \mathbf{A}^T \cdot \mathbf{b}$ . See again [5], [16], [4] for details on the following methods, which we use in a more direct way, as we fix the (maximal) number of iterations to 1000:

1. *Gauss-Seidel iteration.* The iteration formula reads

$$x_i^{*(k+1)} = x_i^{*(k)} + \frac{\varepsilon}{a_{ii}} \cdot \left( b_i - \sum_{j=1}^{i-1} a_{ij} x_j^{*(k+1)} - \sum_{j=i}^n a_{ij} x_j^{*(k)} \right) \quad (12)$$

with  $\varepsilon = 1$ .

2. *Incomplete relaxation.* The speed of the convergence of the Gauss-Seidel method, which tries to minimize (= relax) the defect vector  $\mathbf{b} - \mathbf{A} \cdot \mathbf{x}^*$  completely, might be improved by over-relaxation with  $\varepsilon > 1$ . However, the relaxation factor  $\varepsilon$  has to be chosen carefully; it depends heavily on the system of equations under study.
3. *Jacobi rotation.* This method uses the same kind of rotation matrices as Givens rotations to “improve” the matrix  $\mathbf{A}^T \cdot \mathbf{A}$  by making it diagonally dominant. The grade of this dominance is specified by the rotation limit and the rotation stops as soon as the desired grade is reached. In this work, the algorithm stops in any case after 1000 iterations. Finally, a direct method delivers the solution.

All these methods are sensitive to rounding errors.

### 4.3 Implementation Issues

A proper numerical evaluation of the eigensystem has to be ensured in order to keep the input error for the solution of the system of linear equations (6) as small as possible. Therefore, the eigenvalues should be obtained with great precision which is no problem for on-off sources, even if a numerical search based on (4) was performed. But also the eigenvectors  $\boldsymbol{\varphi}_q$  should be determined with care even if the orders of magnitude of certain elements are extremely small. Section 5.3 demonstrates what happens if the precision of the eigenvector calculation drops. Note that due to  $\lim_{d_i \rightarrow \pm 0} z_i = \mp \infty$ , states with small drifts might cause numerical under-/overflow in (3). But (7) shows that the contribution of such components becomes arbitrarily small. Hence, they can be extracted before solving the system of linear equations (6). As floating-point variables with **double** precision range from about  $\pm 10^{-323}$  to about  $\pm 10^{308}$ , we deleted such equations whose presence would probably lead to overflow in column  $q$  due to  $\exp(z_q K) > 10^{300}$  or to zeros due to  $\exp(z_q K) < 10^{-300}$ . After this reduction, a number of  $n'_S$  equations is left.

As public and commercial tools for numerical mathematics mostly do not allow the user to look inside their implementation, we wrote own code for the solution methods in C++ with great care to avoid bad numerical surprises. To avoid excessive memory consumption and executing times (*c.f.* section 5.3), floating-point variables with **double** precision (8 Bytes) were used instead of such ones with **long double** precision (16 Bytes).

## 5 Results

In this section, we present the quality of numerical results for loss probabilities in quite large systems with homogeneous and heterogeneous sources, which have been obtained on computers of the types Sun SparcStation 10 and 20 (architecture sun4m). Results that were obviously wrong, e.g. negative values, have been marked with “—”.

In relation to the mean burst sizes of the on-off sources under consideration, buffer sizes are classified as follows:

- 1. **XS**: Very small buffer that has no influence on burst level.
- 2. **S**: Small buffer of 0.1 times the minimal mean burst size.
- 3. **M**: Medium-sized buffer of the minimal mean burst size.
- 4. **L**: Large buffer of 10 times the minimal mean burst size.
- 5. **XL**: Very large buffer of 100 times the minimal mean burst size.

In case 1, numerical results might be compared with very stable results provided by the buffer-less fluid flow model. Therefore, results delivered by the fluid flow analysis are regarded to be good enough if the corresponding relative error does not surpass 0.1 %. However, in the cases 2 to 5, fluid flow simulations had to be used in order to obtain reference values. Here, the error tolerance was set to the 95 % confidence interval, whose size was mostly less than 4 % of the corresponding average value. A method for solving the system of linear equations (6) is considered as applicable for one of the following systems if the corresponding error tolerance is kept.

### 5.1 Homogeneous System

The homogeneous system under study consists of  $N \in \{50, 100, \dots, 600\}$  quite bursty on-off sources with a mean-to-peak bit rate ratio of 0.1. These sources are much more bursty and thus more critical than the voice model in [7]. The capacity of the multiplexer was chosen in a way that the load equals 0.8; the size of the state space is  $n_S = N + 1$ .

**XS buffer:** Table 1 compares results obtained by using one direct and two iterative methods with reference results obtained from the buffer-less analysis. The

**Table 1.** Loss probabilities obtained with some solution methods, given homogeneous sources and a very small buffer.

$N$	Full pivot.	Gauss-Seidel	Incompl. relax. (factor)	Jacobi/Buffer-less
100	$3.7389 \times 10^{-2}$	$3.6235 \times 10^{-2}$	$3.6955 \times 10^{-2}$ (1.9)	$3.7389 \times 10^{-2}$
200	$1.3768 \times 10^{-2}$	$1.3532 \times 10^{-2}$	$1.3711 \times 10^{-2}$ (1.9)	$1.3768 \times 10^{-2}$
300	$6.6872 \times 10^{-3}$	$6.6069 \times 10^{-3}$	$6.6875 \times 10^{-3}$ (1.9)	$6.6872 \times 10^{-3}$
400	$3.4852 \times 10^{-3}$	$3.4590 \times 10^{-3}$	$3.4851 \times 10^{-3}$ (1.4)	$3.4852 \times 10^{-3}$
500	$1.9875 \times 10^{-3}$	$1.9799 \times 10^{-3}$	$1.9894 \times 10^{-3}$ (1.2)	$1.9875 \times 10^{-3}$
600	—	$1.1475 \times 10^{-3}$	$1.1475 \times 10^{-3}$ (1.0)	$1.1480 \times 10^{-3}$

**Table 2.** Usefulness of the solution methods given homogeneous sources and a very small buffer.

Gauss without pivoting	★ ★ ★ ★ ★ ★ ★ ★ ★
Gauss with partial pivoting	★ ★ ★ ★ ★ ★ ★ ★ ★ ★ ★
Gauss with full pivoting	★ ★ ★ ★ ★ ★ ★ ★ ★ ★ ★
Givens rotations	★ ★ ★ ★ ★ ★ ★ ★ ★ ★ ★
Householder	★ ★ ★ ★ ★ ★
Gauss-Seidel	
Incomplete relaxation	
Jacobi rotation	★ ★ ★ ★ ★ ★ ★ ★ ★ ★ ★
Number of sources	100 200 300 400 500 600

Gaussian elimination with full pivoting, which behaves best among the direct methods, reproduces the reference values up to 550 sources, but then fails. The Householder method already fails if the number of sources exceeds 300. The Gauss-Seidel method produces better results the more the size of the system increases. The incomplete relaxation method leads to further improvement, but only if the relaxation factor has been optimized before. As there are reference values available, we chose the best result for each  $N$  based on relaxation factors  $\in \{1.0, 1.1, \dots, 1.9\}$ . The Jacobi method with a rotation limit of 0.9 reproduced the reference values exactly for any number of sources, no matter which direct method is used at its end. For a number of 600 sources, only the iteration methods are still applicable. Table 2 summarizes the cases in which the different methods produced results within the defined error tolerance of 0.1 %; these cases are marked by “★”.

**M buffer:** Table 3 shows some results obtained with direct methods and compares them with simulation results.

For 550 sources, the Gaussian elimination without pivoting deviates obviously, while Givens rotations is the only method that manages 600 sources with acceptable precision. As in the previous case, the Householder method is not able to treat more than 300 sources. Among the iterative methods, most of the Gauss-Seidel results (not shown) lie within the corresponding confidence interval. Due to the lack of reference values, i.e. the possibility to optimize the relaxation factor, the incomplete relaxation method is not taken into account. No matter which direct method is used afterwards, the Jacobi rotation does not allow to

**Table 3.** Loss probabilities obtained with some solution methods given homogeneous sources and a medium-sized buffer.

$N$	Gauss	Full pivot.	Givens rot.	Simulation
300	$3.5543 \times 10^{-3}$	$3.5544 \times 10^{-3}$	$3.5544 \times 10^{-3}$	$(3.5701 \pm 0.1202) \times 10^{-3}$
350	$2.5045 \times 10^{-3}$	$2.5044 \times 10^{-3}$	$2.5044 \times 10^{-3}$	$(2.4981 \pm 0.0864) \times 10^{-3}$
400	$1.7992 \times 10^{-3}$	$1.7980 \times 10^{-3}$	$1.7980 \times 10^{-3}$	$(1.7819 \pm 0.0553) \times 10^{-3}$
450	$1.3130 \times 10^{-3}$	$1.3133 \times 10^{-3}$	$1.3134 \times 10^{-3}$	$(1.3034 \pm 0.0550) \times 10^{-3}$
500	$9.6838 \times 10^{-4}$	$9.7026 \times 10^{-4}$	$9.7302 \times 10^{-4}$	$(9.7176 \pm 0.3259) \times 10^{-4}$
550	$7.6164 \times 10^{-4}$	$7.2428 \times 10^{-4}$	$7.2230 \times 10^{-4}$	$(7.0691 \pm 0.2103) \times 10^{-4}$
600	—	—	$5.4560 \times 10^{-4}$	$(5.2717 \pm 0.2138) \times 10^{-4}$

**Table 4.** Usefulness of the solution methods given homogeneous sources and a medium-sized buffer.

Gauss without pivoting	★ ★ ★ ★ ★ ★ ★ ★ ★ ★
Gauss with partial pivoting	★ ★ ★ ★ ★ ★ ★ ★ ★ ★ ★
Gauss with full pivoting	★ ★ ★ ★ ★ ★ ★ ★ ★ ★ ★
Givens rotations	★ ★ ★ ★ ★ ★ ★ ★ ★ ★ ★ ★
Householder	★ ★ ★ ★ ★ ★
Gauss-Seidel	★ ★ ★ ★ ★ ★ ★ ★
Jacobi rotation	★ ★ ★ ★ ★ ★ ★ ★
Number of sources	100 200 300 400 500 600

**Table 5.** Usefulness of the solution methods given homogeneous sources and a large buffer.

Gauss without pivoting	★ ★ ★ ★ ★ ★ ★ ★ ★
Gauss with partial pivoting	★ ★ ★ ★ ★ ★ ★ ★ ★
Gauss with full pivoting	★ ★ ★ ★ ★ ★ ★ ★ ★ ★ ★
Givens rotations	★ ★ ★ ★ ★ ★ ★ ★ ★ ★ ★ ★
Householder	★ ★ ★ ★ ★ ★
Gauss-Seidel	★
Jacobi rotation	
Number of sources	100 200 300 400 500 600

treat more than 400 sources. The results are summarized in Table 4, where “★” stands for a result that lies within the corresponding confidence interval.

**L buffer:** For L buffer, the behavior of the direct methods is quite similar to that in the M buffer case. We confine ourselves to the summary that is presented in Table 5. The Gauss-Seidel method shows larger deviations as in the M buffer case; merely the result for 300 sources lies within the corresponding confidence interval, while the method fails for more than 350 sources. The Jacobi rotation was not able to produce any meaningful result. For L buffer, the direct methods seem to be much better applicable than the considered iteration methods.

5.2 Heterogeneous System

Here, we assume two groups of 30 on-off sources, each with the same mean bit rate and different parameters as follows:

- 1. Mean-to-peak bit rate ratio of 0.5 (less bursty), ratio of buffer and mean burst size  $K/hb_1 \in \{0; 1, 10, 100\}$ ;
- 2. Mean-to-peak bit rate ratio of 0.1 (more bursty), ratio of buffer and mean burst size  $K/hb_2 \in \{0; 1, 10, 100\}$ .

For XS buffer, each direct method is able to compute the corresponding reference value, but table 6 reveals that the Gauss-Seidel method delivers results dependent on the burst sizes, which does not hold for the underlying buffer-less model. This behavior might be interpreted as a signal for the Gauss-Seidel method’s sensitivity to input errors.

Table 7 shows results for the Gaussian methods for different buffer sizes. The result for XL buffer is too small to be compared with simulation results. A comparison with the upper bound for the loss probability that is based on the dominant eigenvalue [6], [2] excludes the results obtained with the Gaussian method without and with partial pivoting; the same is valid for Givens rotations results that are not shown explicitly. As this upper bound is known to overestimate the real result by several orders of magnitude, the result obtained with the Gaussian method with full pivoting seems reasonable. This method seems to be best among the direct methods when dealing with small probabilities. On the other hand, Householder and Gauss-Seidel fail in cases of large buffer sizes. Table 8 shows a summary.

### 5.3 The Role of the Eigenvectors

In this section, we show to which extent the accuracy of the eigenvectors directly influences the numerical stability of the solution methods. Here, only the homogeneous systems need to be investigated (in the heterogeneous case, the eigenvectors are obtained as a composition of parts that are determined for homogeneous groups). Since in this homogeneous case closed formula descriptions of the eigenvalues exist, they can be computed with great precision. However, during the calculation of the eigenvectors, huge differences in the orders of magnitude occur: there are products that contain terms causing numerical overflow as well as numerical underflow while the resulting product would lie in the tractable range concerning the order of magnitude. In our system, the use of floating-point numbers with `double` precision (8 Bytes) within that critical product restricted the number of sources to 100.

A first improvement was reached by a separate treatment of base and exponent of eigenvector elements in two floating-point numbers with `double` precision, i.e. a kind of logarithmic calculation within the critical product. Using this trick, the number of sources could be raised to 250. The price to be paid: On average, the calculation of the eigenvectors took more than 10 times as long. However, the results for up to 600 sources that were reported in the previous section have been obtained using `long double` floating-point numbers (16 Byte). Compared to the use of the standard `double` data type, the speed went down by a factor of more than 100. In general, the effect of input errors on numerical stability seem to grow as the buffer becomes larger. Due to (3), this is not surprising: The eigenvectors are multiplied by exponential functions containing the

**Table 6.** Loss probabilities obtained with the Gauss-Seidel method for heterogeneous sources with different mean burst sizes and very small buffer.

$hb_1$	$hb_2$	Gauss-Seidel	Reference
10	10	$2.4288 \times 10^{-3}$	$2.4376 \times 10^{-3}$
10	100	$2.4260 \times 10^{-3}$	$2.4376 \times 10^{-3}$
100	10	$2.4489 \times 10^{-3}$	$2.4376 \times 10^{-3}$
100	100	$2.4289 \times 10^{-3}$	$2.4376 \times 10^{-3}$

**Table 7.** Loss probabilities obtained with Gaussian elimination methods for heterogeneous sources and M to XL buffers.

$\frac{K}{hb_1}$	$\frac{K}{hb_2}$	No pivoting	Partial pivot.	Full pivot.	Sim./Approx.
1	1	$4.2648 \times 10^{-4}$	$4.2648 \times 10^{-4}$	$4.2648 \times 10^{-4}$	$(4.2132 \pm 0.1059) \times 10^{-4}$
10	10	$1.1415 \times 10^{-7}$	$1.1415 \times 10^{-7}$	$1.1415 \times 10^{-7}$	$(1.1240 \pm 0.0819) \times 10^{-7}$
100	100	$1.0778 \times 10^{-18}$	$7.5006 \times 10^{-19}$	$1.5120 \times 10^{-37}$	Approx.: $1.2992 \times 10^{-33}$

**Table 8.** Usefulness of the solution methods for different buffer sizes given heterogeneous sources.

Gauss without pivoting	★	★	★	
Gauss with partial pivoting	★	★	★	
Gauss with full pivoting	★	★	★	★
Givens rotations	★	★	★	
Householder	★	★		
Gauss-Seidel	★	★		
Buffer size:	XS	M	L	XL

buffer size. Especially iteration methods seem to suffer from this problem; even the powerful Jacobi rotation method fails. The *Gaussian elimination with full pivoting* seems to be the best possible method to get along with this problem.

6 Conclusions

In this paper, we presented some recipes on how to stabilize the stochastic fluid flow analysis numerically if finite buffers and the classical spectral method are used. The most essential step is to determine the eigensystem as precisely as possible. This should be done to keep the input error for the next step low, which is the solution of a system of linear equations in order to adapt the solution of the system of differential equations to the boundary conditions. Before solving that system, those equations that belong to states with very small drift values should be extracted. These states have no significant influence on the solution but may cause numerical under- or overflow. To solve the system of linear equations, the solution method has to be chosen very carefully. Considering the list of possible solution methods that were tested in this work, the use of the Gaussian algorithm with complete pivoting led to the best results, followed by the Givens rotations method. Although the summary tables presented in section 5 are valid only for the corresponding parameter settings, they reflect the experience we also made with other systems. Altogether, the proposed measures lead to a numerical performance of the stochastic fluid flow model analysis that is much better than its reputation. Future work on numerical problems in the fluid flow model context should also consider the matrix-analytic solution by Ramaswami [14] as well as more complicated source models.

## References

1. D. Anick, D. Mitra and M. M. Sondhi. Stochastic theory of a data-handling system with multiple sources. *The Bell System Technical Journal*, 61(8):1871–1894 (1982).
2. A.I. Elwalid and D. Mitra. Effective bandwidth of general Markovian traffic sources and admission control of high speed networks. *IEEE/ACM Transactions on Networking*, 1(3):329–343 (1993).
3. A.I. Elwalid and D. Mitra. Statistical multiplexing with loss priorities in rate-based congestion control of high-speed networks. *IEEE Transactions on Communications*, 42(12):2989–3002 (1994).
4. *Encyclopaedia of mathematics*. Kluwer, 1995, ISBN 1-55608-010-7.
5. D. K. Faddeev and V. N. Faddeeva. *Computational methods of linear algebra*. Freeman, 1963.
6. R. Guérin, H. Ahmadi and M. Naghshineh. Equivalent capacity and its application to bandwidth allocation in high-speed networks. *IEEE Journal on Selected Areas in Communications*, 9(7):968–981 (1991).
7. H. Heffes and D.M. Lucantoni. A Markov modulated characterization of packetized voice and data traffic and related statistical multiplexer performance. *IEEE Journal on Selected Areas in Communications*, 4(6):856–867 (1986).
8. L. Kosten. Stochastic theory of a multi-entry buffer. *Delft Progress Report, Series F*, 1:10–18 (1974).
9. L. Kosten. Stochastic theory of data handling systems with groups of multiple sources. *Proc. of IFIP WG 7.3 – TC 6 International Symposium on the Performance of Computer Communication Systems*, Zürich, 1984, pp 321–331.
10. K.P. Kontovasilis and N.M. Mitrou. Bursty traffic modelling and efficient analysis algorithms via fluid-flow models for ATM IBCN. *Annals of Operations Research*, 49:279–323 (1994).
11. P. Lancaster and M. Tismenitzky. *The Theory of Matrices, Second edition with Applications*. New York: Academic Press, 1985.
12. D. Mitra. Stochastic theory of a fluid flow model of producers and consumers coupled by a buffer. *Advances in Applied Probability*, 20:646–676 (1988).
13. R. Nagarajan, J.F. Kurose and D. Towsley. Approximations techniques for computing packet loss in finite-buffered voice multiplexers. *IEEE JSAC*, 9(3):368–377 (1991).
14. V. Ramaswami. Matrix analytic methods for stochastic fluid flows. *Proc. ITC-16*, Edinburgh, 1999, pp 1019–1030.
15. T. Stern and A. Elwalid. Analysis of separable markov-modulated rate models for information-handling systems. *Advances in Applied Probability*, 23:105–139 (1991).
16. W.J. Stewart. *Introduction to the Numerical Solution of Markov Chains*. Princeton University Press, 1994, ISBN 0-691-03699-3.
17. R. Tucker. Accurate method for analysis of a packet-speech multiplexer with limited delay. *IEEE Transactions on Communications*, 36(4):479–483 (1988).
18. T. Yang and D.H.K. Tsang. A novel approach to estimating the cell loss probability in an ATM multiplexer loaded with homogeneous on-off sources. *IEEE Transactions on Communications*, 43(1):117–126 (1995).



# QoS Support for Real-Time Applications Using the Integration of RSVP/Intserv and Diffserv: A Testbed Experiment

Seong-Ho Jeong<sup>1</sup>, Myung Choi<sup>1</sup>, Randal Abler<sup>1</sup>, Henry Owen<sup>1</sup>, John Copeland<sup>1</sup>,  
and Joachim Sokol<sup>2</sup>

<sup>1</sup> School of Electrical and Computer Engineering, Georgia Institute of Technology,  
Atlanta, Georgia 30332-0250, USA

<sup>2</sup> Siemens AG, Corporate Technology,  
D-81730 Munich, Germany

**Abstract.** The Integrated Services (Intserv) architecture with a resource reservation protocol such as RSVP is able to provide end-to-end quality of service (QoS) to real-time applications. However, it seems that this end-to-end model alone will not scale well for large network topologies. To overcome this scalability problem, the Differentiated Services (Diffserv) architecture has been created, which is based on the control of aggregated traffic. Diffserv, however, does not guarantee end-to-end QoS for real-time applications. Recently, lots of attention is being paid to the integration of RSVP/Intserv and Diffserv because it can provide both end-to-end QoS and scalability. This paper presents a testbed realization of the integration of RSVP/Intserv and Diffserv to provide resource reservation and end-to-end QoS to real-time applications such as voice over IP. We also present some experimental results.

## 1 Introduction

Recent research on QoS support in the Internet has led to two distinct approaches: the Integrated Services architecture (Intserv)[2] and its accompanying signaling protocol, RSVP[3], and the Differentiated Services architecture (Diffserv)[1]. The goal of Intserv is to allow end-to-end QoS to be provided to applications. The current set of services of the Intserv architecture consists of guaranteed and controlled load services. The Intserv architecture needs some explicit setup mechanism to convey information to routers so that they can provide the requested services to flows that require them. RSVP is the most common example of such a setup mechanism. RSVP is a signaling protocol used to request resources from the network. The network responds by explicitly admitting or rejecting RSVP requests [3, 4].

The current prevailing model of RSVP usage is based on a combined RSVP/Intserv architecture where RSVP signals per-flow resource requirements to network elements by using Intserv parameters as defined in the appropriate Intserv service specification

[2]. These network elements apply Intserv admission control to signaled requests. In addition, traffic control mechanisms in each network element are configured to ensure that each admitted flow receives the service requested in strict isolation from other traffic. However, the use of per-flow state and per-flow processing in the RSVP/Intserv approach raises scalability concerns for large networks.

On the other hand, Diffserv has been motivated by the market need for immediate deployment of a QoS solution for the Internet as well as enterprise networks. In contrast to the per-flow orientation of RSVP, Diffserv networks classify packets into one of a small number of aggregated classes, based on the Diffserv codepoint (DSCP) in the packet's IP header [1]. At each Diffserv router, packets are subjected to a per-hop behavior (PHB), which is invoked by the DSCP. A PHB is defined to provide a specific forwarding behavior at each router within the Diffserv domain. Diffserv eliminates the need for per-flow state and per-flow processing and therefore scales well to large networks. However, Diffserv does not guarantee end-to-end QoS for QoS-sensitive applications.

RSVP/Intserv and Diffserv can be considered as complementary technologies in the pursuit of end-to-end QoS and scalability [4]. RSVP/Intserv enables hosts to request per-flow, required resources, along end-to-end data paths and to obtain feedback regarding admissibility of these requests. Diffserv enables scalability across large networks. Therefore, the integration of Intserv/RSVP and Diffserv can provide a combination of scalability and end-to-end QoS.

This paper presents a testbed realization of the integration of RSVP/Intserv and Diffserv to provide end-to-end QoS to real-time applications such as voice over IP. We also present some experimental results. The rest of the paper is organized as follows. In section 2, we describe service definitions and service mapping options between Intserv services and Diffserv services. Section 3 presents our physical testbed structure for the integration of RSVP/Intserv and Diffserv. Section 4 shows experimental results based on different network scenarios. Finally, section 5 presents a summary of the paper and proposed future work.

## 2 Service Definitions and Mapping

The Intserv architecture specifies two service classes: guaranteed service (GS) and controlled-load service (CLS). The GS provides an assured level of bandwidth, a firm end-to-end delay bound and no queueing loss for conforming packets of a data flow. It is intended for applications with stringent real-time delivery requirements.

Unlike guaranteed service, CLS provides no firm quantitative guarantees. If the flow is accepted for controlled-load service, the router makes a commitment to offer the flow a service equivalent to that seen by a best-effort flow on a lightly loaded network. The important difference is that the controlled-load flow does not noticeably deteriorate as the network load increases. By contrast, a best-effort flow will experience progressively worse service as the network load increases. Controlled-load service is intended for those classes of applications that requires a certain level of bandwidth and can tolerate a certain amount of loss and delay.

On the other hand, network elements within the Diffserv region select a PHB as the specific forwarding treatment to provide a specific service to incoming packets. In this paper, we consider three types of forwarding behavior: expedited forwarding (EF), assured forwarding (AF), and default forwarding for best-effort (BE) service.

The goal of the EF PHB is to provide a low loss, low latency, low jitter, assured bandwidth, and end-to-end service through the Diffserv region. Such a service appears to the endpoints like a point-to-point connection or a virtual leased line [10].

In order to create such a service, it is necessary to bound rates such that, at every transit node, the EF aggregate's maximum arrival rate is less than that EF aggregate's minimum departure rate. To do this, it is necessary to configure nodes so that the aggregate has a well-defined minimum departure rate regardless of the intensity of other traffic at the node. The EF PHB provides such a forwarding treatment for a particular Diffserv aggregate. To minimize the impact that EF traffic could have on other traffic, traffic that exceeds a certain limit must be discarded. This maximum EF rate, and burst size if appropriate, must be settable by either static or dynamic means. Several types of queue scheduling mechanisms may be employed to provide the EF PHB. In our testbed, we use class-based queueing (CBQ) [13].

The AF PHB group is a means for the Diffserv region service provider to offer different levels of forwarding assurances for IP packets received from the customer in an Intserv region [11]. Four AF classes are defined in [11], where each AF class in each Diffserv node is allocated a certain amount of forwarding resources such as buffer space and bandwidth. The details of this allocation are subject to a provider/provider or customer/provider SLS and may vary in different domains. IP packets that wish to use the services provided by the AF PHB group are assigned by the customer or the Diffserv region provider into one or more of these AF classes according to the services that the customer has subscribed to. To implement the AF PHB group, we use CBQ.

Service requests in the Intserv region specify an Intserv service type and a set of quantitative parameters known as a *flowspec*. For seamless operations, requests in the Intserv region need to be mapped onto the underlying capabilities of the Diffserv region. To do this, an appropriate PHB, or set of PHBs, for the requested service must be selected, and appropriate policing at the edges of the Diffserv region must be performed. Further, admission control on the Intserv requests that takes into account the resource availability in the Diffserv region should be performed.

After selecting a specific service mapping, the actual mapping of traffic parameters from the Intserv traffic specification to a specification suitable for the Diffserv domain needs to be defined. The specification in the Diffserv domain depends on the Diffserv service type, service level specification (SLS), and admission control procedures. Policy information and the amount of requested/available resources also need to be considered.

The GS requires that the Diffserv region can provide bounded delay and no queuing loss. The ingress edge router needs to know the maximum delay, and the advertised delay parameters in the *adspec* object of the RSVP PATH messages needs to be updated. The EF PHB is a natural choice to support the GS. In this case, however, an end-to-end characterization of the EF PHB in a Diffserv network is necessary to support end-to-end QoS. Further investigation of this mapping is under study.

The CLS could be supported by AF PHB or EF PHB since there are no stringent QoS guarantees to be provided by the network. For AF PHB, the service provisioning or admission control procedures should make sure that the required throughput can be provided to target applications.

Based on the assumption that initial deployment of Diffserv is probably based on static provisioning, we also focus on the static case in the first stage for the mapping of GS-to-EF and CLS-to-AF. Despite the fact that other service mappings are possible, we used this straightforward mapping.

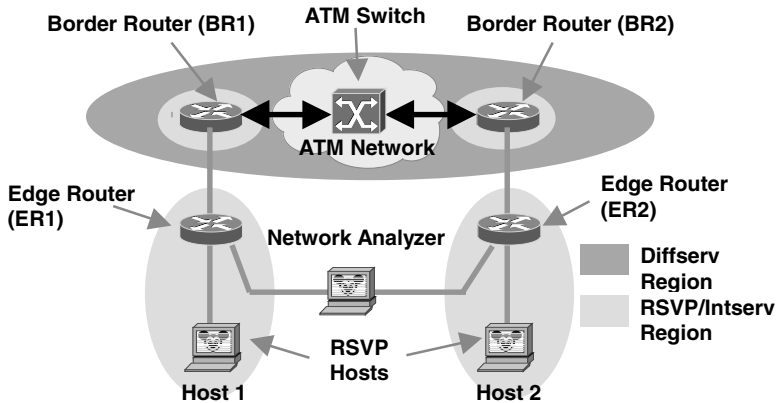


Fig. 1. Physical testbed for the integration of RSVP/Intserv and Diffserv

### 3 Physical Testbed for RSVP/Intserv and Diffserv Integration

Our physical testbed in Figure 1 is based on [4], where end-to-end QoS is provided by coupling two Intserv regions with a Diffserv region. Intserv regions are located at the periphery of the network, and the Diffserv region is in the core of the network. The edge routers at the network boundary interface to the Intserv and Diffserv regions. We use ATM as the backbone technology. Although this is not strictly necessary in order to examine the concepts presented in this paper, we believe that the use of ATM backbone will be common due to deployed infrastructure.

The Diffserv region of the network is treated as virtual links connecting Intserv capable routers. Within the Diffserv region of the network routers implement specific PHBs for aggregate traffic control. The total amount of traffic that is admitted into the Diffserv region that will receive a certain PHB is limited by policing at the edge. The Diffserv region of the network is able to support the Intserv style services requested from the periphery.

To focus on the mapping aspect and the behavior of the aggregates within the Diffserv domain, we assume that the Diffserv domain is not RSVP aware. In addition, we

consider a single QoS sender (Host1 or Host2) in one of the Intserv regions and a single QoS receiver (Host1 or Host2) in the other.

The network elements such as routers and hosts in the testbed are based on the Linux operating system. Recent Linux kernel versions offer a wide variety of traffic control functions to support QoS [5, 6]. We utilize those traffic control functions to realize an integration of RSVP/Intserv and Diffserv. The following sections describe the testbed structure in further detail.

### 3.1 Intserv and Diffserv Regions

Each Intserv region consists of an Intserv capable host (Host1 or Host2) and an edge router (ER1 or ER2). A key requirement in the Intserv regions is that the host and edge router need to be able to process RSVP messages.

The Diffserv region supports aggregate traffic control and is assumed not to be able to support per-flow classification. As we mentioned above, devices (mainly routers) in the Diffserv region are not RSVP aware, and therefore they will pass RSVP messages transparently. The Diffserv region provides three levels of PHBs based on the DSCP in packet headers: EF, AF, and BE.

The Diffserv region is able to provide support for the standard Intserv QoS services between its edge routers. It is also possible to invoke these services by use of standard PHBs within the Diffserv region. The Diffserv region provides admission control information to the Intserv regions through static service level specifications enforced at the edges of the Diffserv region. The Diffserv region is able to pass RSVP messages, in such a manner that they can be recovered at the egress of the Diffserv region.

### 3.2 Hosts Within the Intserv Regions

Both sending and receiving hosts (Host 1 and Host 2 in Figure 1) use RSVP to communicate the QoS requirements of QoS-aware applications running on the host. A daemon process called *rsvdpd* is handling all RSVP signaling on behalf of these applications. In our testbed, we are using an implementation of ISI and Alexey Kuznetsov's code [7]. Figure 2 illustrates interactions between the application and *rsvdpd*. The figure also shows the interaction between the host, edge router, and border router.

Despite the fact that the necessary marking of the IP packets with the appropriate DSCP could occur on every node (including the host) before the packet enters the Diffserv domain, we decided to use the edge router for this task, because only traffic leaving the Intserv domain must be marked. On the other hand, host marking seems to be the best way to achieve an appropriate marking from the applications point of view, but this solution suffers from the missing knowledge which service classes are provided by the network.

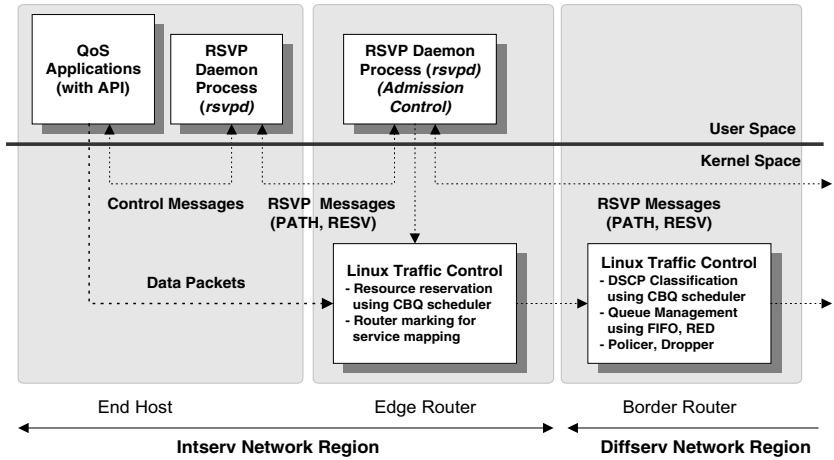


Fig. 2. Interactions between application, *rsvpd* process, and traffic control functions

### 3.3 Edge Routers

Edge routers (ER1 and ER2 in Figure 1) interface to both the RSVP/Intserv region and the Diffserv transit region. Edge routers are able to process PATH and RESV messages. They store RSVP state and provide per-flow classification. Thus, the edge routers are able to identify and process traffic on a per-flow granularity.

The edge router also provides the admission control function based on the transmit capacity which is statically provisioned at each Diffserv service level. To adapt a Diffserv network to new requirements, either reconfiguration of the edge routers or a more novel concept like a bandwidth broker to manage the overall network resources can be used.

In order to request resources from the network, QoS applications in the Intserv region use RSVP as an explicit signaling protocol. These requests will be accepted or rejected by our edge routers based on a local table which specifies the transmit capacity provisioned at each Diffserv service level. The network will send a rejection message in response to requests for resources that would exceed the available capacity. Consequently, the upstream host and QoS application will have the information needed to take corrective action. As a result, the integrity of those flows that were admitted can be preserved, at the expense of the flows that were not admitted. Thus, by appointing an Intserv-conversant admission control agent in the edge router for the Diffserv region of the network it is possible to enhance the service that the network can provide to QoS applications.

### 3.4 Border Routers

Border routers (BR1 and BR2 in Figure 1) reside in the Diffserv region. In our implementation, these routers act as pure Diffserv routers since the Diffserv region is not RSVP aware. As such, their sole responsibility is to police submitted traffic based on the service level specified in the DSCP and the agreement negotiated with the customer (aggregate traffic control). RSVP messages are ignored by these routers in the Diffserv region and then processed at edge routers ER1 and ER2 according to standard RSVP processing rules.

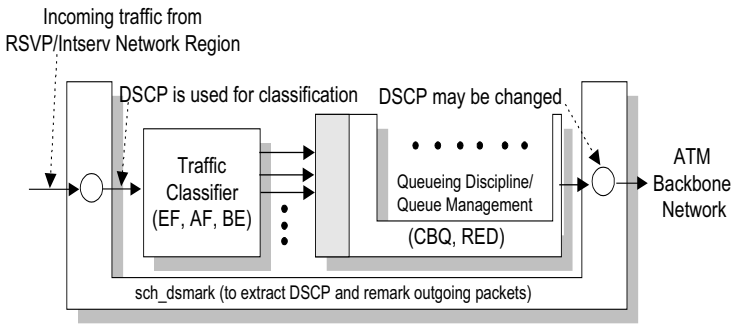


Fig. 3. The internal structure of border routers

The internal structure of border routers is shown in Figure 3, which is based on [6]. Border routers are configured to perform packet classification based on DSCP, packet scheduling, queue management, policing/dropping. The following sections describe some of the functions in further detail.

#### 3.4.1 Packet Classification Based on DSCP

Within the Diffserv region, service is allocated to traffic based on the DSCP in each packet's IP header. Therefore, it is necessary to mark the DSCP correctly to obtain a particular level of service within the Diffserv region. Note that to protect Intserv traffic in Diffserv regions, DSCPs assigned to such traffic should not overlap with the DSCPs assigned to other traffic.

It is desirable to allocate service to traffic based on the application and/or user originating the traffic. Port number and source IP address in the IP headers can be used to identify packets associated with a specific application and a specific user, respectively.

#### 3.4.2 Packet Scheduling

As we mentioned above, we use CBQ as our packet scheduler, which is an approach proposed in [13]. For CBQ a single set of mechanisms is proposed to implement link-sharing and real-time services. The mechanisms include a classifier to classify

arriving packets to the appropriate class, an estimator to estimate the bandwidth recently used by a class, a selector to determine the order in which packets from the various classes will be sent, and a delayer or overlimit action to schedule traffic from classes that have exceeded their link-sharing limits and are contributing to congestion. In our implementation, CBQ is used to classify EF, AF, and BE traffic so that each user can get appropriate resources based on the user service profile.

### 3.4.3 Queue Management

An AF implementation must attempt to minimize long-term congestion with each class, while allowing short-term congestion resulting from bursts. This requires an active queue management algorithm.

We consider here only random early detection (RED) as an active queue management algorithm for routers. In contrast to traditional queue management algorithms, which drop packets only when the buffer is full, the RED algorithm drops arriving packets probabilistically [14]. The probability of drop increases as the estimated average queue size grows. Note that RED responds to a time-averaged queue length, not an instantaneous one. Thus, if the queue has been mostly empty in the “recent past”, RED won’t tend to drop packets unless the queue overflows. On the other hand, if the queue has recently been relatively full, indicating persistent congestion, newly arriving packets are more likely to be dropped. The RED algorithm itself consists of two main parts: estimation of the average queue size and the decision of whether or not to drop an incoming packet. In our testbed, RED is attached to the classes created for BE as well as AF.

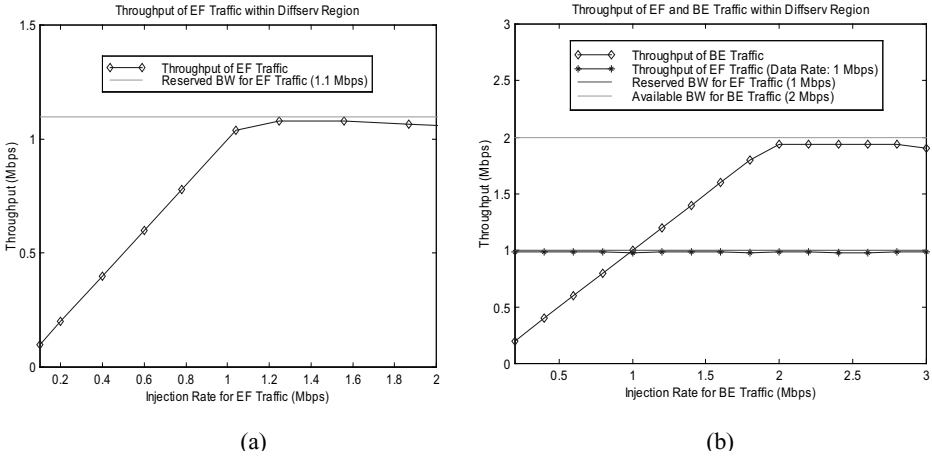
## 3.5 ATM Switch

An ATM switch is the only component of our ATM backbone and provides high-speed connectivity within the Diffserv region. The ATM switch is also used to create congestion in our testbed by assigning a limited bandwidth to a permanent virtual circuit (PVC). To connect the border routers with the ATM switch, we use Classical IP over ATM (CLIP) which is defined in RFC1577.

## 4 Experimental Results

We consider voice-over-IP (VoIP) traffic as a real-time traffic source in our testbed. We are using a network analyzer to generate simulated VoIP traffic. Voice is divided into small time windows, called frames, and the voice samples comprising each frame are encoded for transmission. For transmission of voice flows over IP networks, one or more frames are carried in each IP packet. We call this the packetization of the voice signal. The header bytes often imply a substantial bandwidth overhead for carrying voice over IP networks.





**Fig. 4.** (a) Throughput of EF traffic within the Diffserv region (b) Throughput of EF and BE traffic within the Diffserv region

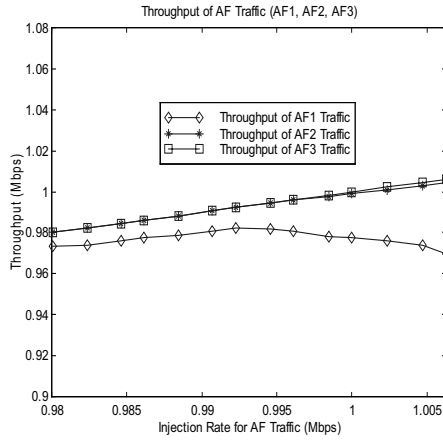
Because packets of a VoIP flow using UDP flow can be dropped by an IP network due to congestion or arrive out of sequence at the destination, an additional protocol called real-time transport protocol (RTP) is necessary to enable receivers to reorder out-of-sequence packets and to detect packets, which are outside the time window for processing. Therefore RTP is adding a sequence number and a timestamp to every packet.

For our experimental studies, we assume that each VoIP packet encodes two 10 ms G.729 frames, for a total of 20 ms of speech. The VoIP packet size is 60 bytes, which includes 20 bytes for the two G.729 frames plus an overhead of 40 bytes (20 bytes for IP header, 8 bytes for UDP header, and 12 bytes for RTP header).

Figure 4-(a) shows how well our testbed can support the EF PHB within the Diffserv region. In this experiment, we use one aggregate traffic source which generates simulated 60-byte VoIP packets marked with EF DSCP (101110). The Diffserv routers (BR1 and BR2 in Figure 1) maintain a separate queue for EF class. The aggregate rate at which the traffic source produces EF-marked packets ranges from 0.1 Mbps to 2.0 Mbps. The reserved bandwidth in the Diffserv region for EF-marked packets is 1.1 Mbps. As shown in Figure 4-(a), EF-marked traffic gets the desired throughput performance until the injection rate is approximately 1.1 Mbps. Note that the traffic which exceeds the reserved bandwidth must be discarded in order to minimize the impact that EF traffic could have on other traffic.

Figure 4-(b) shows throughput performance when BE packets and EF-marked packets are transmitted simultaneously over the Diffserv region. Reserved bandwidth for EF-marked traffic is 1 Mbps, and EF-marked packets are generated at the rate of 1 Mbps. The BE packets are marked with BE DSCP (000000). The aggregate rate at which the BE traffic is generated ranges from 0.2 Mbps to 3 Mbps, and available bandwidth for BE traffic is 2 Mbps. RED is used as an active queue management scheme for BE traffic. The goal of this experiment was to explore the impact of BE

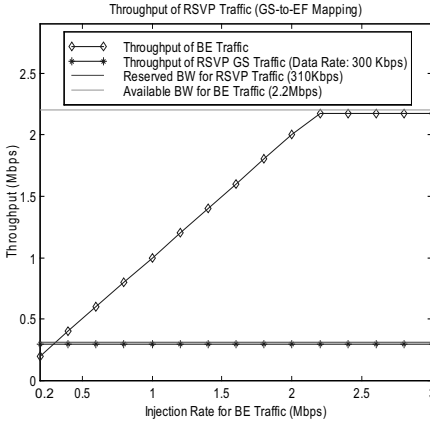
traffic on the EF-marked traffic. As shown in Figure 4-(b), EF-marked traffic gets the desired throughput behavior. Furthermore, the increase in the injection rate for BE traffic does not have any serious impact on the throughput performance of EF-marked traffic.



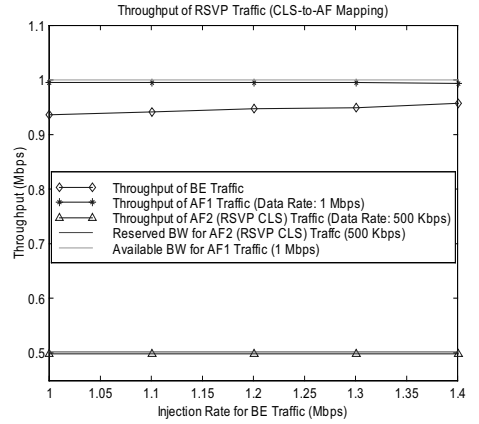
**Fig. 5.** Throughput of AF traffic within the Diffserv region

Figure 5 illustrates the throughput behavior of AF-marked traffic. In this experiment, we consider 3 AF classes, AF1, AF2, and AF3. Aggregate traffic for each class is generated by an independent traffic source and is marked with a corresponding AF DSCP (AF1: 001110, AF2: 010110, AF3: 011110). For simplicity, we only consider a single drop precedence level within each AF class. All traffic sources generate AF-marked packets at the same rate, and the rate ranges from 0.98 Mbps to 1.006 Mbps. The priority levels 5, 4, and 3 are assigned to AF1 class, AF2 class, and AF3 class, respectively. That is, AF3 class is given the highest priority, and AF1 class is given the lowest priority. This means that when packets with different priority levels arrive at the Diffserv router simultaneously, the packets with an assigned lower priority level are served first. RED is used as an active queue management scheme for all AF classes. As shown in Figure 5, the AF1-marked traffic gets the lowest throughput performance as we expected.

As discussed in Section 2, GS can be mapped onto the EF PHB. Figure 6 shows throughput performance of RSVP/Intserv GS traffic when GS-to-EF service mapping is used. The aggregate rate for GS traffic is 300 Kbps and reserved bandwidth for GS traffic is 310 Kbps. Maximum output link capacity of BR1 is artificially limited to 2.5 Mbps. Approximately 2.2 Mbps is available to BE traffic. To examine the impact that the BE traffic could have on the RSVP/Intserv traffic, injection rate for BE traffic ranges from 0.2 Mbps to 3 Mbps. As shown in Figure 6, the RSVP/Intserv traffic gets the desired throughput performance. Furthermore, an increase in the injection rate for BE traffic does not have any serious impact on the RSVP/Intserv traffic.



**Fig. 6.** Throughput of RSVP/Intserv GS traffic (GS-to-EF mapping)



**Fig. 7.** Throughput of RSVP/Intserv CLS traffic (CLS-to-AF mapping)

Similarly, the AF PHB can be used to support CLS. Figure 7 shows the throughput performance of RSVP/Intserv CLS traffic when a CLS-to-AF service mapping is used. In this experiment, we create two AF classes in the Diffserv region: AF1 and AF2. We also create a BE class for best-effort traffic. AF2 class is given higher service priority than AF1 and BE classes. To protect CLS traffic from other traffic, CLS is mapped onto the AF2 class. The aggregate data rate for CLS traffic is 500 Kbps, and the reserved bandwidth for the CLS traffic is 500 Kbps. The aggregate data rate for AF1-marked traffic is 1 Mbps, and the available bandwidth for AF1-marked traffic is 1 Mbps. The maximum output link capacity of BR1 is deliberately limited to 2.5 Mbps. To explore the impact that the BE traffic has on the CLS traffic, the injection rate for BE traffic ranges from 1 Mbps to 1.4 Mbps. As shown in Figure 7, CLS traffic gets the requested throughput. In addition, an increase in the injection rate for BE traffic does not have any serious impact on the CLS traffic. Compared to BE traffic, AF1-marked traffic gets better performance as expected.

## 5 Conclusions and Future Work

This paper presented a testbed realization of the integration of RSVP/Intserv and Diffserv. There is still a great deal of research underway to determine how Integrated and Differentiated Services should and will interoperate. In the work presented here we used a mapping of guaranteed service into the expedited forwarding per-hop behavior. We also used a mapping of controlled-load service into the assured forwarding per-hop behavior, and best-effort traffic was marked with a default differentiated service code point.

In our Linux-based testbed implementation, we used class-based queueing to classify traffic so that each user got the appropriate resources. We also used random early detection as our active queue management algorithm for the assured forwarding and best effort classes. We used a SmartBits 2000 traffic generator to generate simulated time-sensitive voice-over-IP packets. We then made measurements of the performance of the testbed. We used Pentium II-based 300MHz machines.

Now that we have an interoperating integrated and differentiated service testbed, we plan to investigate other possible mapping options, dynamic service provisioning, enhanced admission control procedures, and detailed performance issues.

## Acknowledgements

This work was partially supported by grants from Hitachi Telecom and Siemens.

## References

1. Blake, S. et al: An Architecture for Differentiated Services. RFC 2475, December 1998.
2. Braden, R., Clark, D., Shenker, S.: Integrated Services in the Internet Architecture: an Overview. Internet RFC 1633, June 1994
3. Braden, R., Zhang, L., Berson, S., Herzog, S., Jamin, S.: Resource Reservation Protocol (RSVP) Version 1 Functional Specification. RFC 2205, Proposed Standard, Sep. 1997
4. Bernet, Y., Yavatkar, R., Ford, P., Baker, F., Zhang, L., Speer, M., Braden, R., Davie, B., Wroclawski, J., Felstaine, E.: A Framework for Integrated Services Operation over DiffServ Networks. Internet Draft, draft-ietf-issll-diffserv-rsvp-03.txt, Sep. 1999
5. Almesberger, W., et al: Linux Traffic Control - Implementation Overview. <http://lrc-www.epfl.ch/linux-diffserv/>, Nov. 1998.
6. Almesberger, W., Salim, J.H., Kuznetsov, A.: Differentiated Services on Linux. Internet Draft, draft-almesberger-wajhak-diffserv-linux-01.txt, June 1999.
7. University of Southern California (USC)/Information Sciences Institute (ISI): RSVP Software. <http://www.isi.edu/div7/rsvp/release.html>.
8. Guerin, R., Blake, S., Herzog, S.: Aggregating RSVP based QoS Requests. Internet Draft, draft-guerin-aggreg-rsvp-00.txt, Nov. 1997.
9. Nichols, K., et al: Definition of the Differentiated Services Field (DS Field) in the IPv4 and IPv6 Headers. RFC 2474, Dec. 1998.
10. Jacobson, V., Nichols, K., Poduri, K.: An Expedited Forwarding PHB. IETF RFC 2598, February, 1999
11. Heinanen, J., Finland, T., Baker, F., Weiss, W., Wroclawski, J.: Assured Forwarding PHB Group. IETF RFC 2597, Feb. 1999.
12. Nichols, K., Jacobson, V., Zhang, L.: A Two-bit Differentiated Services Architecture for the Internet. Internet Draft, draft-nichols-diff-svc-arch-01.txt, Apr. 1999.
13. Floyd, S., Jacobson, V.: Link-sharing and Resource Management Models for Packet Networks. IEEE/ACM Transactions on Networking, Vol. 3, No. 4, pp. 365-386, Aug. 1995.
14. Floyd, S., Jacobson, V.: Random Early Detection gateways for Congestion Avoidance. IEEE/ACM Transactions on Networking, Vol.1, No.4, pp. 397-413, Aug. 1993.

# Efficient End-to-End Transport of Soft Real-Time Applications\*

Zoe Antoniou<sup>1</sup> and Ioannis Stavrakakis<sup>2</sup>

<sup>1</sup> Nokia Research Center  
3 Burlington Woods Drive, Burlington, MA, 01803, USA  
`zoe.antoniou@nokia.com`

<sup>2</sup> Department of Informatics, University of Athens  
Athens, 15784, Greece  
`istavrak@di.uoa.gr`

**Abstract.** Typically, real-time applications are well supported in environments where there is ample bandwidth available, enabling their transport within the stringent time constraints. Accordingly, non-real-time applications are well supported in environments where there is ample storage available resulting in low losses. For an emerging class of applications such as prerecorded video - referred to as soft real-time applications there may be an increased role that storage can play - in addition to bandwidth - in their effective transport. A scheme is proposed in this paper that takes advantage of available storage within the network and the end users, to efficiently transport prerecorded continuous media traffic across a number of network nodes.

The scheme is based on a scheduling policy implemented within the network nodes called Deadline Credit (DC) policy. Its objectives are (a) to utilise all (if possible) of the available capacity by allocating bandwidth *dynamically* to a number of data streams, (b) to distribute the available bandwidth *fairly* between all competing data streams and (c) to maximise the number of Application Data Units (ADUs) that are served by always giving priority to the one with the shortest time to expiration. Numerical results are presented and the induced ADUs loss rates under the proposed policy are compared against those of some alternative schemes.

## 1 Introduction

The transmission of continuous media traffic is an inherently difficult problem due to the time sensitive nature and the traffic variability of these applications [1], [2].

Consider, for instance, the transmission of video applications in which it is required that frames (images) be transmitted at a certain rate in order to

---

\* This work is part of Zoe Antoniou's Ph.D. research while with Northeastern University and has been supported by the National Science Foundation under Grant NCR 9628116.

guarantee no starvation or overflow at the receiving end. Bits generated over a frame form a logical block of information (Application Data Unit or ADU) which is central to both the traffic generation process at the transmitter and the reconstruction process at the receiver. Such a role may be played by a smaller encoding unit than the frame such as a macroblock [3]. Since frames are typically generated at a constant frame rate the nature of such an application is periodic.

ADUs can be mapped into a collection of network data units or ATM cells. In order to guarantee a timely reconstruction at the receiver, complete frames should be delivered at a constant display rate. This implies that all cells of an ADU should be delivered in time for the reconstruction process in order to avoid either starvation or buffer overflow at the receiving end. When network resources are reserved at peak demand, all cells of the ADU will be delivered on time, leading to a timely and error-free reconstruction at This is clearly the case when resources are over-allocated to a number of applications to increase network utilisation.

The requirement for timely delivery of information associated with such continuous media traffic suggests that deadlines could be defined and deadline-driven policies be designed. If the nature of an application requires that a complete ADU should be received before it can be processed by the receiver, then it would be reasonable to associate a common deadline with all the cells in the same data unit and develop scheduling policies that aim to maximise the number of complete data units that are delivered by their respective due date. This approach is different from many traditional approaches in which every cell is assigned its own deadline and the scheduling policy typically attempts to maximise the number of individual cells transmitted by their due date [4], [5].

Continuous media traffic may be real time or prerecorded such as Video on Demand (VoD). In the prerecorded case, which is the focus of this paper, there can be various alternatives to the efficient transmission of the data since the information is available *a priori*, as opposed to the real time traffic where real time processing and transmission is the only approach. In either case, they both have a periodic *consumption* rate once the application starts running.

The proposed policy, called Deadline Credit (DC) policy, focuses on the efficient scheduling of soft real-time constrained applications. Its first objective is to utilise all (if possible) of the available capacity by allocating bandwidth *dynamically* to a number of (continuous media) data streams. Secondly, it aims to distribute the available bandwidth *fairly* between all competing data streams. Finally, it aims to maximise the number of ADUs that are served by always giving priority to the one with the *shortest time to expiration*. In order to achieve this the DC policy uses all of the available bandwidth to (a) serve the ADUs with upcoming deadlines and (b) send ADUs in advance in order to build deadline credit. This deadline credit can be *consumed* later when the demand for resources may increase.

In this paper, the DC approach is described for a multihop network for the case of prerecorded traffic. It is an extension of the DC policy described in [6]. For time critical applications, such as continuous media applications, the most

important resource is high bandwidth availability in order to achieve timely delivery of information. Traditionally, large buffer space has not been very useful in this environment. Bandwidth becomes an even more scarce resource when the demand increases and the system is utilised near its full capacity.

In the case of the DC approach, however, large buffers can help improve the performance of soft real-time constrained applications. When the demand for bandwidth is low ADUs can be sent in advance to the downstream nodes in the process of building up the deadline credit of the stream (assuming buffer availability). Naturally, the larger the available buffer space the higher is the potential for the deadline credit build up. In turn, the higher the deadline credit build up the higher the number of ADUs which are “forwarded” in advance towards the destination. Also, the more links the ADUs traverse in advance the closer to the destination they reach.

Consequently, large buffers can be useful in forwarding as *many* ADUs as possible, as *close* to the destination as possible *in advance* in an attempt to exploit the available bandwidth when the demand is low. This way, congestion is expected to be alleviated or avoided at a later time when the demand may increase. This is the driving motivation behind applying the DC approach to a multihop network.

In [7] an overview of some related policies is presented. Among others, the problem of scheduling and transmitting prerecorded traffic and, in particular, video has been addressed in [3], [?] where a policy called Constant-Rate Transmission and Transport (CRTT) was proposed. According to CRTT, a number of frames (ADUs) are sent in advance (prior to the commencement of playback). This provides an initial “build up” and enables the remaining frames to be transmitted at a constant rate during playback. The amount of “build up” and the rate of transmission are computed given the characteristics of the movie which are known a priori and the amount of buffering available at the receiving end. However, the issue of multiplexing a number of streams carrying prerecorded traffic or how the losses will be distributed between streams sharing the same link for high network utilisation is not directly addressed as it is for the policy proposed here. In addition, the allocation of network resources is static and is determined once at the beginning of the session, as opposed to the dynamic allocation performed by the DC policy based on the varying network resource requirements from all users. The CRTT policy is rate based and deadlines are not explicitly defined as opposed to the DC policy.

## 2 The DC Policy

In this paper a multihop network of source, intermediate and destination nodes (with bidirectional links) is considered. A node can have a number of incoming and outgoing links. Each link arrives at (or leaves from) a port. Ports for incoming links have an amount of buffer space allocated to the streams that arrive at this port. Ports for outgoing links have a scheduler/server. Streams are multiplexed and transmitted by the scheduler/server onto the correct output link

in order to reach to the corresponding receiver where they are decoded and reconstructed. Source nodes also have a mass storage device and a file server. The issues associated with retrieving streams of data from a storage device are beyond the scope of this paper.

The system is assumed to be slotted; a cell requires a fixed amount of time for transmission referred to as a slot. All departures from the node occur at slot boundaries. Let  $j$  denote a data stream (application);  $j$  takes values in the range  $0 \leq j \leq J - 1$ . A variable with superscript  $j$  represents a quantity associated with stream  $j$ . Let  $n$  denote a node;  $n$  takes values in the range  $1 \leq n \leq N$ . A variable with subscript  $n$  represents a quantity associated with node  $n$ . Also, let  $P_i^j$  denote the  $i^{th}$  ADU from stream  $j$  which corresponds to the  $i^{th}$  periodic interval of the session of stream  $j$ ; let  $\hat{P}_i^j$  represent its length in cells. (Subscript  $n$  is not used with variable  $P$ .)

The per stream (application) traffic is assumed to be bursty in nature. It consists of application data units, ADUs; each ADU is of variable length (measured in cells). For real time traffic a new ADU is assumed to be generated periodically every  $T^j$  slots, where  $T^j$  denotes the period of the generation process of the application measured in slots. The  $i^{th}$  ADU is generated at the  $i^{th}$  periodic interval. So for real time traffic the number of ADUs available to the server at any time is limited by the generation process rate of the application. For prerecorded traffic though, all of the ADUs of the session are available at the source node prior to the commencement of transmission.

The application is assumed to have a periodic reconstruction rate; an ADU has to be available to the receiver every  $T^j$  slots for processing ( $T^j$  is assumed to be both the generation and the reconstruction periodic interval of the application). So each application has a maximum allowable end-to-end delay,  $ETE^j$ , which is defined to be the same for all ADUs of stream (application)  $j$ . This  $ETE^j$  delay is the sum of all propagation delays between nodes, and the sum of all queuing and transmission delays (called node delays in this paper) at each node for stream  $j$ . Propagation delays depend on link capacities, physical distance between nodes and the number of hops. If the sum of all propagation delays is subtracted from  $ETE^j$ , the remaining is the total amount of time an ADU can spend on queuing and transmission delays in all the nodes. This amount of time can be distributed in a variety of ways between nodes. For instance, it can be distributed equally between nodes or based on their utilisation. The distribution can be static (once upon connection set-up) or dynamic (varying based on the load of each node during the session). As it will be clarified in section 2.1, in this paper the node delays are defined cumulatively from one node to the next. If part of the node delay is not used in one node it is available to be used at a later node.

The QoS metric considered here is the ADU loss probability. An ADU is assumed to be lost if any one of its cells is lost. The DC policy is *work conserving* in the sense that it will serve as long as there are ADUs available for service and buffer space is available at the next downstream node.



The proposed scheduling algorithm determines when the next stream is to be selected for service, which stream is to be selected and whether an ADU from the selected stream is to be served or dropped. The scheduling algorithm is executed at every outgoing port of each node. A set of metrics is used in making the above decisions, namely, the deadline credit, the losses and the priority index of the stream. A counter is associated with each metric per node  $n$  and stream  $j$ : (a) the deadline credit counter,  $D\_CR_n^j$ , (b) the losses counter,  $L\_CR_n^j$  and (c) the priority index counter,  $P\_CR_n^j$ . In addition to the three metrics two more parameters are used in making scheduling decisions: (d) the sequence number,  $SN$  and (e) the buffer availability.

A sequence number is defined for each ADU of stream  $j$ . The definitions of the counters and parameters and their descriptions are given in the following subsections, which are followed by a description of the scheduling procedure. A more detailed description can be found in [7].

It is assumed that key events occur and actions are taken instantaneously at the end of slots. If during a slot either the last cell of an ADU or no cell is being served, the server will be idle at the end of this slot and this is referred to as a **decision slot**. At decision slots certain actions are taken leading to the selection of a stream to be served next. For the selected stream this decision slot becomes an **examination slot**; ADUs of stream  $j$  can be scheduled for service or dropped only at examination slots of stream  $j$ . The losses counter of a stream is updated only at the examination slots of this stream as it will be discussed later.

The ADU **at the head of stream  $j$**  is defined to be the ADU of stream  $j$  which has the earliest due date among all ADUs of stream  $j$  not served or dropped yet. As it will become clear later, expired ADUs can either remain in the system until the next examination slot for the corresponding stream at which time they are dropped or they can be dropped upon expiration and all the relevant counters can be updated.

## 2.1 The Deadline Definition

An end-to-end delay,  $ETE^j$ , is defined for each stream  $j$ . This consists of the total propagation delay from source to destination (from node 1 to node 2,  $PROP\_DELAY_{1,2}^j$ , from node 2 to node 3,  $PROP\_DELAY_{2,3}^j$ , and so on), plus the maximum allowed queuing and transmission delay at each node (for node 1  $NODE\_DELAY_1^j$ , for node 2  $NODE\_DELAY_2^j$  and so on). The sum of all propagation, queuing and transmission delays is equal to the end-to-end delay. So ADU  $P_i^j$  is needed at the receiver  $ETE^j$  slots after the instant marking the beginning of the  $i^{th}$  periodic interval of stream  $j$  or  $(i \cdot T^j + ETE^j)$  slots after the beginning of the session of stream  $j$ .

Using the assigned node and propagation delay tolerances a deadline is defined per stream and node. The deadline is defined cumulatively from one node to the next such that if an ADU leaves a node before the expiration of its deadline by an amount of  $t$  slots, these slots are automatically available at later nodes in addition to their default deadline/delay tolerance where they may be

used if there is higher contention for bandwidth. If an ADU can not meet its deadline at any node it is dropped.

The deadline of  $P_i^j$  is defined as the number of time slots from the instant marking the beginning of the  $i^{th}$  periodic interval of stream  $j$  as follows:

- At node 1:  $D_1^j = NODE\_DELAY_1^j$
- At node 2:  $D_2^j = NODE\_DELAY_1^j + PROP\_DELAY_{1,2}^j + NODE\_DELAY_2^j$   
 $= D_1^j + PROP\_DELAY_{1,2}^j + NODE\_DELAY_2^j$
- At node 3:  $D_3^j = D_2^j + PROP\_DELAY_{2,3}^j + NODE\_DELAY_3^j$
- At node  $n$ :  $D_n^j = D_{n-1}^j + PROP\_DELAY_{n-1,n}^j + NODE\_DELAY_n^j$

Dropping an ADU at a node  $n$  due to the node deadline violation,  $D_n^j$ , does not imply that the  $ETE^j$  delay would have been violated as well. There is always the possibility that this ADU could have met the  $ETE^j$  delay if it experienced delays at later nodes that were less than their respective deadlines. In that case, the ADU would have been dropped unnecessarily but this is a consequence of the distribution of the end-to-end deadline among the involved nodes.

he  $ETE^j$ .

The **Additional Delay Tolerance** (ADT) of an ADU of stream  $j$  is defined to be equal to the amount of remaining time until the ADU expires reduced by its default deadline  $D_n^j$ . That is if ADT is equal to  $d\_cr$  for an ADU of stream  $j$ , this ADU will expire after  $d\_cr + D_n^j$  slots.

## 2.2 The Sequence Number

The SN marks the sequence of ADUs of a particular stream  $j$ . The SN is stored in the header of the ADU. It is used to identify which ADU corresponds to which periodic interval in order to determine whether the ADU can meet its deadline or has expired and needs to be dropped.

More specifically, if an ADU can not meet its deadline at any node  $n$  it will be dropped and the next ADU will be scheduled for service. The scheduler can determine the number of expired ADUs at the waiting queue of the current node from the value of the deadline credit counter of the node. In the downstream nodes however, the value of their deadline credit counter is not enough to accurately determine the number of ADU that were dropped at upstream nodes (if any). It is in this case that the scheduler needs to read the SN of the incoming ADUs to determine how many ADUs have been dropped at upstream nodes (if any) and make the necessary adjustments to all the relevant counters. The SN could be used in all cases to identify the sequence of ADUs but the above paragraph explained the case where it would be necessary. Its use will be clarified further after reading sections 2.4 and 2.5.

## 2.3 The Buffer

Ports for incoming links have an amount of buffer space allocated to the incoming streams. Buffer space may be assigned separately to each stream or collectively

a set of streams. As part of the scheduling process each node, e.g.  $n$ , needs to know the amount of available buffer space of the next downstream node, e.g.  $n + 1$ .

In this paper, it is assumed that buffer space is assigned separately per stream at each incoming port. This guarantees that when the buffer condition (introduced in section 2.4) holds no ADUs will be dropped due to buffer overflow. For implementation purposes, the buffer availability can be checked in the following way. Each node  $n$  knows the total amount of buffer space available per stream at the next downstream node,  $n + 1$ , referred to as  $Buffer_{n+1}^j$ . In addition, each node keeps a record of the total amount of data transmitted per stream measured in cells or bytes or any other appropriate unit,  $data_n^j$ . Finally, each node requires from each downstream node that it transmits data to the amount of data it has transmitted, again measured per stream, referred to as  $data_{n+1}^j$ . The buffer availability per stream in the next downstream node can be determined as follows:

$$Available\_buffer_{n+1}^j = Buffer_{n+1}^j - (data_n^j - data_{n+1}^j)$$

The above formula will account for all the ADUs currently in the waiting queue of node  $n + 1$  as well as all the ADUs that are currently traveling towards node  $n + 1$ .

If buffer space were assigned collectively for the set of streams arriving at a node  $n + 1$  from the same node  $n$  then it would still be possible to guarantee no buffer overflow using the above method. If, however, the buffer pool at a node  $n + 1$  is shared between streams arriving from different nodes then guaranteeing no buffer overflow may not be possible or it may require additional feedback of information between nodes, thus, increasing the processing overhead.

The amount of buffer space available at the next node at any instant determines how many ADUs can be sent in advance. Feedback is necessary to update upstream nodes on the value of the  $data_n^j$  parameter. It is assumed that feedback is available periodically, based on which a stream can build a maximum possible deadline credit from the current value of  $D\_CR_n^j$  to  $D\_CR_n^j\_MAX$  till the next buffer availability update. It is assumed that the buffer of the next node  $n + 1$  can store a maximum of  $b$  data units according to the latest buffer feedback update. This corresponds to a session duration of  $bT^j$  slots. So the difference between the current value of  $D\_CR_n^j$  and the current value of  $D\_CR_n^j\_MAX$  represents the remaining buffer space of the corresponding receiving buffer based on the latest feedback update. In the following sections,  $D\_CR_n^j\_MAX$  is used to define the buffer condition when making scheduling decisions.

## 2.4 The Deadline Credit Counter

The purpose of the deadline credit counter is to keep track of the ADT of the ADU at the head of the corresponding stream. At the same time, the deadline credit counter indicates how far ahead or behind schedule is the service of the ADUs of the corresponding stream still in the system. ADUs do not need to carry

the value of the deadline credit counter as they propagate from one node to the next. A deadline credit counter is defined for each stream  $j$  and node  $n$  that stream  $j$  traverses, denoted by  $D\_CR_n^j$ . The counter is initialised and updated such that upon arrival of a new ADU at a node  $n$ , the value of  $D\_CR_n^j$  is the same as the value of  $D\_CR_{n-1}^j$  when transmission of the ADU was completed at node  $n-1$ . This enables the scheduler at any node  $n$  to determine how far ahead or behind schedule is the service of the ADUs of stream  $j$  *in all the nodes up to and including* node  $n$  and use the value of the counter in making scheduling decisions.

At the beginning of session  $j$ , deadline credit counters are initialised at all the nodes that stream  $j$  will traverse. The counter of each node is initialised such that the propagation delays and maximum allowable node delays of upstream nodes are accounted for. For instance, suppose that stream  $j$  traverses nodes 1 through  $N$ . The deadline credit counters at each node are initialised as follows:

- Node 1:  $D\_CR_1^j = 0$
- Node 2:  $D\_CR_2^j = D\_CR_1^j + NODE\_DELAY_1^j + PROP\_DELAY_{1,2}^j$
- Node 3:  $D\_CR_3^j = D\_CR_2^j + NODE\_DELAY_2^j + PROP\_DELAY_{2,3}^j$
- Node  $n$ :  $D\_CR_n^j = D\_CR_{n-1}^j + NODE\_DELAY_{n-1}^j + PROP\_DELAY_{n-1,n}^j$

This is consistent with the deadline definitions as shown in section 2.1 The deadline credit counters are initialised cumulatively from one node to the next such that if the first ADU leaves a node before the expiration of its deadline by an amount of  $t$  slots, these slots are automatically available at later nodes in addition to their default deadlines.

During the session  $D\_CR_n^j$  is updated as follows:

- (a) During every elapsed slot:  $D\_CR_n^j = D\_CR_n^j - 1$ . This step captures the fact that any slot that passes by reduces the ADT of the ADU at the head of the queue of stream  $j$  by one time slot.
- (b) At the end of an examination slot for stream  $j$ :  $D\_CR_n^j = D\_CR_n^j + (k+m)T^j$  where  $k$  is all the dropped ADUs (which are all expired) and the binary (0/1) variant  $m$  is 1 only if : (i)  $D\_CR_n^j + (k+m)T^j \leq D\_CR_n^j\_MAX$  (buffer condition), and (ii)  $D\_CR_n^j + (k+m)T^j + D_n^j \geq \hat{P}_i^j$  (deadline condition), where  $P_i^j$  is the non expired ADU at the head of stream  $j$ .  
Step (b) “adjusts” the context of  $D\_CR_n^j$  to describe the ADT of the ADU at the head of stream  $j$  after removing the dropped/scheduled for service ADUs of stream  $j$ . Step (b-(i)) checks the buffer availability and step (b-(ii)) checks the deadline condition for the ADU at the head of stream  $j$ .

## 2.5 The Losses Counter

In addition to the  $D\_CR_n^j$  counter, a second counter,  $L\_CR_n^j$ , is set up to count the number of ADUs that are dropped from stream  $j$  up until the last examination slot for stream  $j$ .  $L\_CR_n^j$  is updated at examination slots for stream  $j$  only as follows:

$$L\_CR_n^j = L\_CR_n^j + k$$

where  $k$  is equal to the number of ADUs of stream  $j$  dropped in the current examination slot for stream  $j$ . The dropped ADUs may belong to two categories: either ADUs that expired while waiting at node  $n$ , or ADUs that were dropped at an earlier node because their deadline could not be met.

As it was mentioned earlier, in the first case the value of the deadline credit counter at node  $n$  is sufficient to determine the number of expired ADUs. In the second case, the scheduler uses the SN. In both cases, the losses counter,  $L\_CR_n^j$ , is updated with the losses for all the nodes up to and including node  $n$  because: (a) losses that occur at previous nodes can be detected through the SN, and (b) losses that occur at node  $n$  can be detected through the  $D\_CR_n^j$  or the SN.

## 2.6 The Priority Index Counter

In order to guarantee that the available bandwidth will be distributed fairly between all competing streams, the two metrics are combined into a single metric, called priority index, that aims to capture both the current deadline credit of the stream and its cumulative losses. The priority index counter  $P\_CR^j$  is defined

$$P\_CR_n^j = \frac{D\_CR_n^j - T^j \cdot L\_CR_n^j}{T^j}$$

The priority index counter represents the relative priority of competing data streams. The smaller the value of the priority index counter of a stream the higher the priority of the stream with respect to the others. This will be the stream with the most losses, if any, and among streams with no losses or the same losses it will be the stream with the lowest deadline credit.

## 2.7 The Scheduling Algorithm

The proposed algorithm is summarised for any node  $n$  in the following flowchart. A more detailed and insightful description can be found in [7].

## 3 Numerical Results and Discussion

The performance of the DC policy is investigated under two different scenarios which demonstrate the improvement in the induced ADU loss rate between the DC policy and an alternative approach which permits no deadline credit build up. The alternative approach schedules only one ADU per period subject to buffer availability. In the following graphs, the ratio of the induced ADU losses is plotted as a function of the number of multiplexed streams.

The average bandwidth required by all streams,  $U_{req}$ , is defined as the sum of the average cell rate generation of all streams. So for a total of  $N$  data streams:

$$U_{req} = \sum_{j=0}^{N-1} \frac{\hat{P}_{ave}^j}{T^j}$$

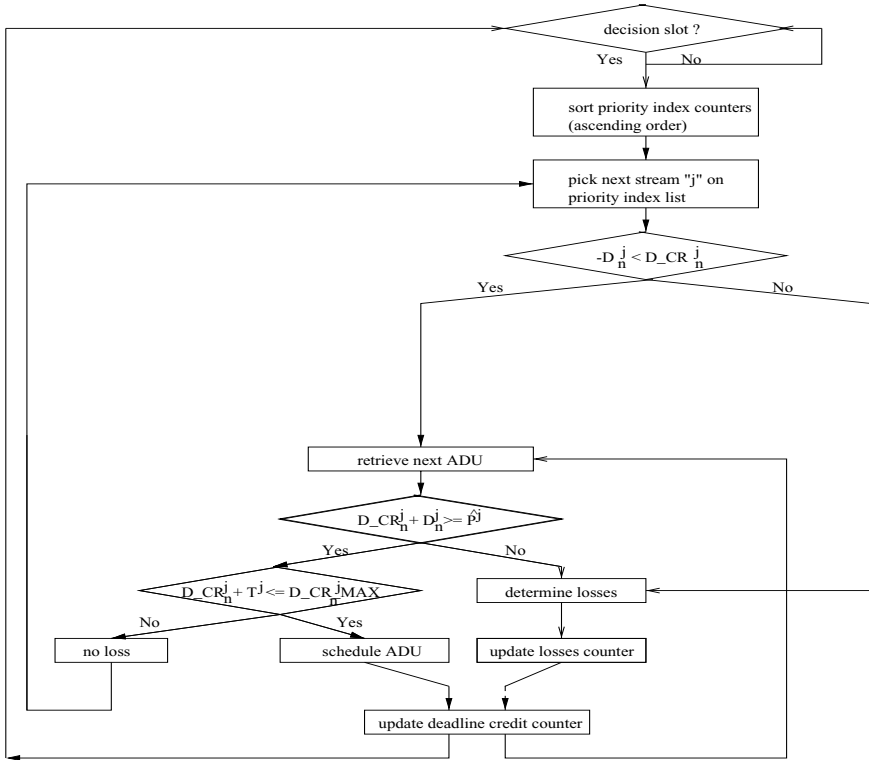
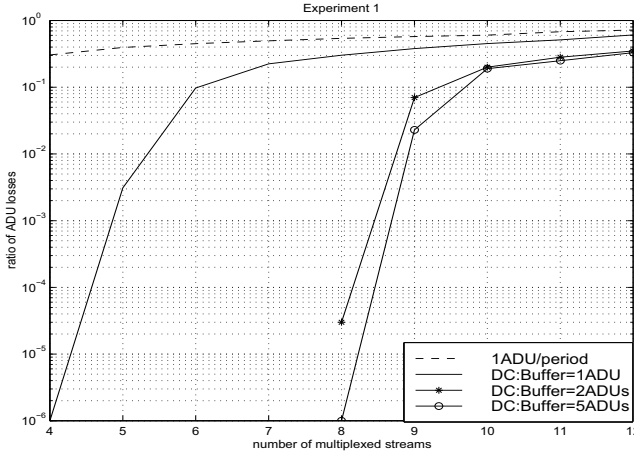


Fig. 1. The DC scheduling algorithm flowchart.

### 3.1 Experiment 1

The first scenario considers the case where a set of streams has the same source (node 1), destination (node 3) and the same intermediate nodes (node 2; only one intermediate node in this experiment). Its purpose is to show that the DC algorithm utilises the available bandwidth and buffer space and improves on the induced ADU losses as compared to the alternative policy. The intermediate and destination nodes are assumed to have the same amount of available buffer space per stream. The node delays are distributed equally per node.

All streams, except from two background traffic streams, are assumed to have the same traffic characteristics. They have the same period,  $T^j = 100$  slots, the same ADU length distribution (uniform with maximum length  $LP_{max} = 20$  cells), the same end-to-end deadline which is equal to one period,  $ETE^j = T^j$  slots, and the same QoS requirements. The two background traffic streams have the same characteristics as above with the exception that the length of ADUs has a bursty distribution. It is equal to 1 cell with probability  $p$  and equal to  $LP_{max} = 20$  with probability  $1 - p$ . For this experiment  $p = 0.3$ , which implies that the background traffic streams generate long ADUs 70% of the time.



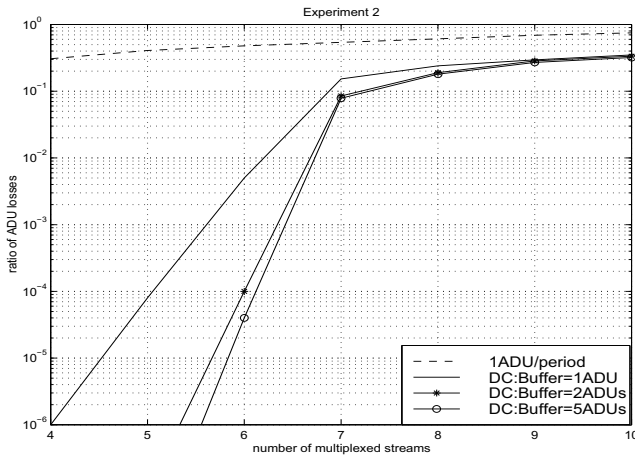
**Fig. 2.** Experiment 1: Ratio of induced ADU losses vs number of multiplexed streams.

In Fig. 2, the ratio of ADU losses under the DC policy is plotted for the cases where the available buffer space at each node is at most equal to 1ADU, 2 ADUs and 5ADUs of maximum length per stream. Each stream requires an average of 10.5% of the link bandwidth; for example, 5 streams correspond to 52.5% of each link's utilisation.

For the DC policy, the case of  $Buffer = 1ADUs$  is not very indicative of its performance as it allows for very limited deadline credit build up. With respect to the alternative approach of serving one ADU per period, there is a significant difference in the losses, especially for the cases of  $Buffer = 2ADUs$  and  $Buffer = 5ADUs$ . For system utilisation approaching 100% the losses rates converge. This is expected since there is very little or no bandwidth available to be used for building deadline credit and, as a result, the available buffer space can not be utilised. So the losses are comparable in all cases.

### 3.2 Experiment 2

The second scenario considers the case where a set of streams with homogeneous traffic characteristics has to travel from the same source (node 1) to the same destination (node 3) through one intermediate node (node 2). In addition, there are background traffic streams (in this case three streams) which travel from node 1 to node 2. At node 2, they can be assumed to terminate or continue on to another link, separately from the rest of the streams. Their purpose is to consume bandwidth over the first hop in an ON-OFF fashion. As in experiment 1, they are assumed to generate ADUs of length equal to 1 cell with probability  $p = 0.5$  and equal to  $LP_{max} = 20$  with probability  $1 - p = 0.5$ . When in the OFF state they allow the DC policy to build deadline credit. When in the ON state they create a bottleneck for the rest of the streams over the first hop.



**Fig. 3.** Experiment 2: Ratio of induced ADU losses vs number of multiplexed streams (not including background traffic).

The objective of this scenario is to show that the DC algorithm dynamically builds deadline credit by utilising the available bandwidth and buffer space at nodes 2 and 3.

Fig. 3 shows the ratio of the induced ADU losses versus the number of multiplexed streams (not including the background traffic streams). Over the first hop (node 1 to node 2) the total number of streams is always the number of streams indicated in the x-axis plus three background traffic streams. Over the second hop (node 2 to node 3) the total number of streams is the number indicated on the x-axis. As in Experiment 1, there is a significant improvement in the induced ADU losses between the two approaches and for the case of the DC policy, the performance improves with increasing available buffer space.

## References

1. Klara Nahrstedt Ian R. Philp and Jane W.S. Liu. Scheduling and Buffer Management for Soft-Real-Time VBR Traffic in Packet-Switched Networks. Technical Report, *University of Illinois at Urbana-Champaign, Ill*, 1996.
2. George Kesidis. *ATM Network Performance*. Kluwer Academic Publisher, 1996.
3. J.M. Mc Manus and K.W. Ross. Video on Demand over ATM: Constant-Rate Transmission and Transport. *Technical Report, University of Pennsylvania, Philadelphia, PA*, <http://www.seas.upenn.edu/~ross/>, November 1995.
4. Zoe Antoniou and Ioannis Stavrakakis. Earliest Due First Scheduling for Application-Level QoS Delivery. *Proceedings of IEEE Conference on Protocols for Multimedia Systems - Multimedia Networking*, pp.172-182, Santiago, Chile, November 1997.
5. Sujata Banerjee. Translating Application Requirements to ATM Cell Level Requirements. *Proceedings of IEEE International Conference on Communications*, 1997.



6. Z. Antoniou and I. Stavrakakis. Deadline Credit Scheduling Policy for Pre-recorded Sources. To appear in *Proceedings of GLOBECOM '99*, December 5-9 1999.
7. Z. Antoniou and I. Stavrakakis. Efficient End-to-End Transport of Soft Real Time Applications. *Report, CDSP Centre, Northeastern University*, 1999.
8. J.M. Mc Manus and K.W. Ross. Prerecorded VBR Sources in ATM Networks: Piecewise-Constant-Rate Transmission and Transport. *Technical Report, University of Pennsylvania, Philadelphia, PA*, <http://www.seas.upenn.edu/~ross/>, September 1995.

# Real-Time Traffic Transmission over the Internet

Marco Furini<sup>1,\*</sup> and Don Towsley<sup>2,\*\*</sup>

<sup>1</sup> Department of Computer Science, University of Bologna,  
Mura Anteo Zamboni 7, 40127 Bologna, Italy  
`furini@cs.unibo.it`

<sup>2</sup> Department of Computer Science, University of Massachusetts,  
Amherst MA 00103, USA  
`towsley@cs.umass.edu`

**Abstract.** Multimedia applications require the transmission of real-time streams over a network. These streams often exhibit variable bandwidth requirements, and require high bandwidth and guarantees from the network. This creates problems when such streams are delivered over the Internet. To solve these problems, recently, a small set of differentiated services has been introduced. Among these, Premium Service is suitable for real-time transmissions. It uses a bandwidth allocation mechanism (BAM) based on the traffic peak rate. Since the bandwidth requirement of a video stream can be quite variable, this can result in a high cost to the user and an inefficient use of network bandwidth. In this paper we introduce a BAM that can increase bandwidth utilization and decrease the allocated bandwidth without affecting the QoS of the delivered real-time stream and without introducing any modification in the Premium Service. We also introduce several frame dropping mechanisms that further reduce bandwidth consumption subject to a QoS constraint when coupled with the above BAM. The proposed BAM and the heuristics algorithms are evaluated using Motion JPEG and MPEG videos and are shown to be effective in reducing bandwidth requirements.

## 1 Introduction

Pay per view movies, distance learning, and digital libraries are examples of multimedia applications that require the transmission of real-time streams over a network. Such streams (such as video) can exhibit significant bit rate variability [6], depending on the encoding system used, and can require high network bandwidth. Moreover these real-time streams require performance guarantees from the network (e.g., guaranteed bandwidth, loss rate, etc.). This poses significant problems when these real-time streams are delivered over the Internet, as the Internet is not currently able to provide any type of guarantee. Although these

---

\* The work of this author was performed while visiting the University of Massachusetts

\*\* The work of this author was funded in part by the National Foundation grant ANI-9805185. Any opinions, finding, or recommendations expressed in this paper are those of the authors and do not necessary reflect the views of the National Science Foundation.

applications are currently deployed in the Internet, the quality of service (QoS) that they receive is far from what is desired.

There has been considerable activity recently on defining and introducing a small set of differentiated services into the Internet in order to improve the service of certain classes of applications. One such proposal by Nichols, et al. [8] introduces a new service, called *Premium Service*, which can provide the QoS required by a real-time stream. Briefly, premium service provides a low loss, bounded low delay, and fixed bandwidth channel (equal to the peak rate associated with the video). However, the peak rate allocation can be expensive in terms of bandwidth and inefficient in terms of bandwidth utilization when the video has been encoded using a variable bit rate (VBR) encoding scheme. This variability coupled with the peak rate bandwidth allocation can lead to high cost and inefficient use of bandwidth.

To reduce the variability of the traffic, smoothing techniques [9] [3] [1] [2] have been introduced. In practice, smoothing produces a new, less variable transmission of the traffic that, although having the same QoS requirement, requires less bandwidth. However, even if smoothing is used, a BAM based only on the traffic peak rate can still lead to low bandwidth utilization and to a waste of bandwidth.

In the past, several attempts have been made to solve the problem of inefficient bandwidth utilization [13] [7] [1] [2]. However, some problems arise with these proposals. For instance, in [13] to increase the bandwidth utilization, the client must settle for a lower QoS; [7] [1] [2] use a dynamic BAM to increase the bandwidth utilization. Unfortunately, this dynamic BAM can result in a temporary disruption of service if additional required bandwidth is unavailable. Hence, the previous proposals cannot be used to efficiently transmit real-time streams with perfect QoS. In the future, it may be possible to solve this problem through the advanced reservation of resources [10].

The first contribution of this paper is the introduction of a new bandwidth allocation mechanism that increases bandwidth utilization and decreases the allocated bandwidth without affecting the QoS of the real-time stream delivered and without requiring any modification to the Premium Service architecture introduced in [8]. For a given video stream, the BAM allocates the peak bandwidth to the premium channel. This allocation is progressively reduced as the peak rate of the remainder of the stream decreases. When coupled with optimal smoothing [9], bandwidth consumption is substantially reduced even though the resulting smoothed stream doesn't look like a decreasing function.

Although the proposed BAM can substantially reduce bandwidth consumption, there may still be a further need to reduce bandwidth consumption. Our second contribution consists of several frame dropping mechanisms that further reduce bandwidth consumption subject to a QoS constraint when coupled with the above BAM. These mechanisms provide the flexibility for the client to negotiate a tradeoff between bandwidth consumption and QoS degradation with the server (and network). Using these heuristics, we show through simulation that it is possible to substantially reduce the bandwidth consumption while drop-

ping only a few frames; depending on the movie, we can save up to 43% of the bandwidth while dropping only 1% of the frames.

The paper is organized as follow. In Section 2 we introduce our bandwidth allocation mechanism. In Section 3 we present the experimental results obtained using our BAM. In Section 4 we present the benefit of delivering slightly imperfect QoS and in Section 5 we present some heuristic algorithms for dropping some frames and the experimental results using these algorithms. The conclusions are presented in Section 6.

## 2 An Efficient Bandwidth Allocation Mechanism

In this section we introduce a new BAM that increases bandwidth utilization and decreases the allocated bandwidth of a stream without affecting its QoS and without requiring any modification to the Premium Service architecture. Premium Service was introduced in [8] in order to provide the QoS required by real-time streams; the server set up a premium service connection with a bandwidth equal to the peak rate associated with the video. The server can do this because the video is completely stored, and so all the video characteristics are known. Due to the variability in a video stream, this mechanism can lead to low resource utilization which may not be acceptable when someone pays for the allocated bandwidth, as is likely to happen in the coming years.

The benefit of our BAM come from the dynamic bandwidth allocation that we use during transmission. Unlike the dynamic BAMs described in [4][7], our mechanism provides the QoS required by the real-time stream. This is because our BAM begins by allocating the peak bandwidth to the premium channel, but progressively reduces the allocation as the peak rate of the remaining stream decreases (this progressive reduction is possible because the bandwidth characteristics of the video are known ahead of time). Consequently, there never is a need to increase bandwidth during the session. For this reason we provide the same guarantee as the classical peak rate BAM, while using less bandwidth.

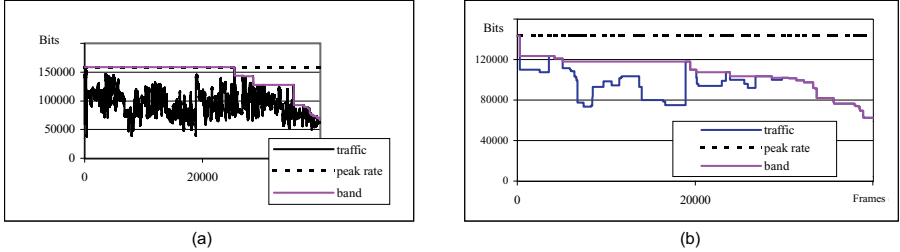
Unfortunately it is very rare that the bandwidth required by the real-time stream is a decreasing function in time. For this reason our BAM is not directly based on the bandwidth requirements but is based on a *bandwidth function*, based on the traffic shape, that will be defined below.

In the following we consider a sender (that provides the service) and a receiver (that desires the service). The receiver requests a video from the sender. This video is composed of  $N$  frames. Without loss of generality, we assume a discrete time-model where one time unit corresponds to the time between successive frames. For a 24 fps full motion video, the duration of a frame is  $1/24^{th}$  of a second. We denote by  $a(i)$  the data sent during time  $i$ , for each  $i = 1, \dots, N$ . We introduce the following *bandwidth function*, which will be used by the BAM:

$$band(i) = \max\{a(j) | j \geq i\} \quad i = 1, \dots, N. \quad (1)$$

Since our BAM uses this non-increasing function, no additional bandwidth will be requested during the transmission and it will also be possible to reduce the

bandwidth when it is no longer needed: at time  $j$  (just before sending the quantity  $a(j)$ ), a request to de-allocate the bandwidth is sent if  $band(j) < band(j-1)$  and the new allocated bandwidth will be  $band(j)$ , instead of  $band(j-1)$ . In Fig. 1, we show the behavior of the *bandwidth* function for a real-time video stream. Fig. 1(a) shows the behavior of the *bandwidth* function for a pure VBR video while Fig. 1(b) shows the behavior for the same movie, but smoothed. We note that our BAM allocates less bandwidth than the peak rate BAM because it decreases the bandwidth when future frames do not need it.



**Fig. 1.** Our BAM versus the classical BAM. (a) Pure VBR traffic. (b) Smoothed traffic.

The bandwidth utilization,  $U$ , achieved using our bandwidth allocation mechanism is  $U = \sum_{i=1}^N a(i) / \sum_{i=1}^N band(i)$  and is greater than what is obtained using the classic BAM because  $band(i) \leq Peak\_rate$   $i = 1, \dots, N$ .

In the next section, we present experimental results obtained using several video traces that illustrate the benefits (like reducing bandwidth and increasing bandwidth utilization giving complete guarantees and QoS) that are possible using the proposed BAM in the premium service architecture.

### 3 Experimental Results

In this section we present experimental results obtained from analyzing several Motion JPEG (MJPEG) and MPEG encoded videos.

#### 3.1 Motion JPEG

The MJPEG algorithm compresses each video frame independently using the JPEG still-picture standard. We use four different MJPEG videos [1], *Big*, *Sleepless in Seattle*, *Crocodile Dundee*, and *ET*, each consisting of 40000 frames (28 minutes). Each video is smoothed [9] considering the client buffer of 1 MB and zero startup delay. These experiments illustrate the benefits of our BAM by showing the reduction in the amount of required bandwidth that it introduces.

In Fig. 2 we show the bandwidth allocation curve *band* and the bandwidth requirements for two of the four MJPEG videos. Also shown is the peak rate allocation. First we note, that even when the traffic is smoothed, it is still quite

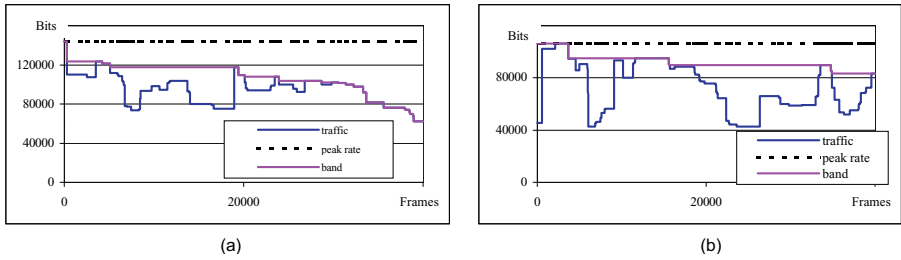


Fig. 2. (a) *Big*. (b) *Sleepless in Seattle*.

variable and that this variability results in the over allocation of bandwidth by a peak rate BAM. Our BAM allocates less bandwidth while still providing the same QoS because *band* better fits the traffic shape than a peak rate allocation. The primary reason that these benefits are so great is that the peak rate of the smoothed traffic occurs in the initial part of the movie. The benefits are less with *ET* and *Crocodile Dundee* because the peak rate occurs late in the video (refer to [5] for a detailed description).

Fig. 3 quantifies the benefits of the proposed BAM over a peak rate allocation. To better compare the two BAMs, we present in Fig. 3(a) the bandwidth saved by implementing our BAM in the premium service architecture. We observe that the reduction in bandwidth ranges from 5% (*ET*) to 25% (*Big*). Fig. 3(b) compares the bandwidth utilization achieved by the peak rate BAM and the proposed BAM. The proposed BAM always increases the bandwidth utilization. In one case, *Big*, the increase is greater than 20%. Finally, we note that the benefits are strongly correlated to the proximity of the traffic peak rate to the beginning of the video.

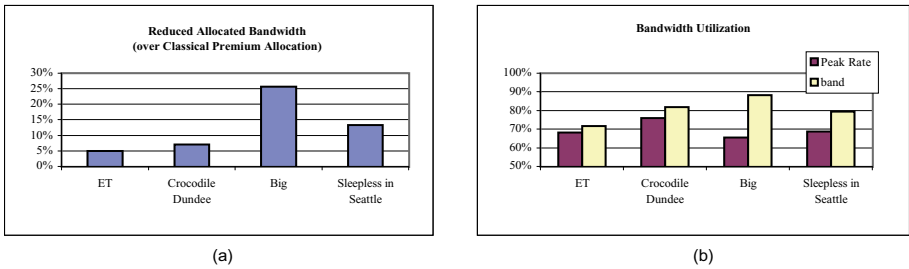


Fig. 3. (a) it shows the amount of bandwidth saved if the proposed BAM is used in the premium service mechanism. (b) it compares the bandwidth utilization reached by the peak rate allocation and the proposed BAM.

### 3.2 MPEG Traces

In this section we present results obtained from analyzing MPEG traces. MPEG is an inter-frame dependency encoding mechanism that has smaller average frame size than the MJPEG encoding. We use four different videos: *MTV*, *The Silence of the Lambs*, *Jurassic Park* and *Starwars*. All traces are 28 minutes long (except *Starwars* that is 121 minutes long), contain 12 frame GOPs and are 24 fps.

We consider a client with 1 MB of buffer available for storing video. We also considered a startup delay of 24 frames (one second). We chose this value in order to obtain considerable smoothing benefits while not incurring too long a startup delay at the client [9].

Fig. 4 shows the allocation using the peak rate BAM and the proposed BAM. We observe that the peak rate BAM wastes a considerable amount of bandwidth while the proposed BAM allocates less bandwidth without compromising the QoS of the video delivered. The reason why the proposed BAM can save a lot of bandwidth is because the traffic peak rate is in the first half of the movie. Due to the lack of space, results obtained from analyzing *Jurassic Park* and *Starwars* are not presented here, but we refer the reader to [5] for a detailed explanation.

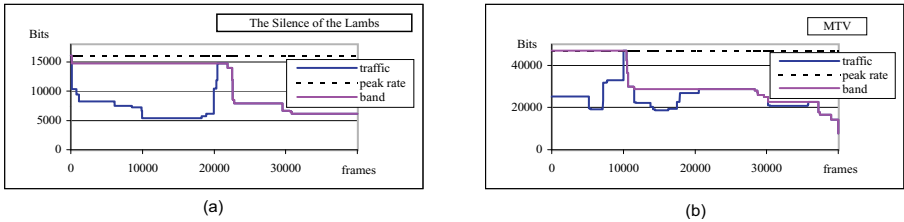
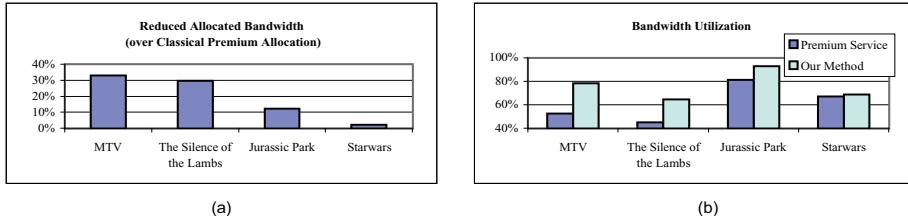


Fig. 4. (a) *The Silence of the Lambs*. (b) *MTV*.

Fig. 5 quantifies the benefits of our proposed BAM over a peak rate allocation. The benefits introduced are remarkable since the reduction in bandwidth requests ranges from 2% (*Starwars*) to more than 30% (*MTV*). *Starwars* yields little benefit because the peak rate of the smoothed traffic occurs very close to the end of the video (see [5] for a detailed description).

### 3.3 Conclusion

In this section we presented the experimental results obtained from analyzing several video traces. We quantified the bandwidth allocated for both MJPEG and MPEG videos by the proposed BAM and the peak rate BAM. The proposed BAM can substantially reduce the required bandwidth for both types of videos without affecting the QoS. If in the coming years, the cost to the customer is proportional to the allocated bandwidth, our mechanism will result in a lower cost for transmitting the same video with the same QoS. Moreover, our BAM doesn't require any modification to the proposed premium service architecture.



**Fig. 5.** (a) It shows the amount of bandwidth that is possible to save if the proposed BAM is used in the premium service architecture. (b) It shows the bandwidth utilization reached by the proposed BAM and the peak rate BAM.

## 4 Benefits in Delivering Imperfect QoS

In the previous sections we presented a BAM for transmitting perfect QoS real-time streams such as video and illustrated its benefits over a peak rate BAM. Although the bandwidth improvement can be substantial, there may be times when a client may require a further bandwidth reduction while being willing to tolerate some degradation in the QoS. Hence, in this section we present several frame dropping mechanisms which, when coupled with our BAM, provides the client and the server with the capability to tradeoff bandwidth utilization with the QoS. Imperfect QoS, does not imply unpredictable QoS. Although less than perfect, the QoS is made known to the client. In fact, the sender and the receiver must agree on this QoS: the server and the receiver should establish a contract in which the receiver agrees to receive a defined non-perfect QoS video and the sender agrees to provide a service with that particular QoS.

Imperfect QoS means that some frames of the video are dropped. If the video is also smoothed, as recommended, the smoothed traffic presents a slightly imperfect QoS video. The client receives a continuous, although imperfect, stream and continues to play it without halting after dealing with the imperfections through loss concealment [12]. There are no changes in the BAM presented in the previous section, because it doesn't matter if the data, used by the *band* function, represents a perfect or a non-perfect QoS.

In [5] we describe an algorithm which minimizes the peak rate of a smoothed video when the client has a constraint on the number of frames that can be dropped and the video was encoded using an intra-frame encoding algorithm. This algorithm relies on the MINFD algorithm developed in [13] that minimizes the number of frames that have to be dropped if the system has both *network bandwidth* and *client buffer* constraints. The MINFD algorithm has been shown to be optimal with respect to the minimum number of frames discarded. A detailed explanation of the MINFD algorithm can be found in [13].

Unfortunately, we are concerned with a different problem: given a specified QoS (e.g. the number of frames to drop), we are interested in reducing the *average allocated bandwidth* and not in minimizing the peak rate of the smoothed video. For this reason we cannot use the MINFD algorithm. Instead it is necessary to develop other algorithms that can reduce the amount of bandwidth needed



for transmitting a video, given a number of frames to drop. We present some heuristic algorithm for doing this in the next section.

## 5 Heuristic Algorithms

In this section, we present several heuristic algorithms that further try to reduce the allocated bandwidth by dropping frames from the video before smoothing it. As we stated earlier, an imperfect QoS should be used in a scenario where the server and the client agree on a particular QoS. For instance, the client could agree to receive video at an imperfect QoS if it could pay less for the same service (one can think of a distance learning systems, where the QoS of the video might not be very important). We propose algorithms that discard a certain number of frames corresponding to the QoS established by the server and the client (let us denote this number by  $k$ ) in order to reduce the bandwidth needed for transmitting the video. We develop separate sets of heuristic algorithms for MJPEG and MPEG.

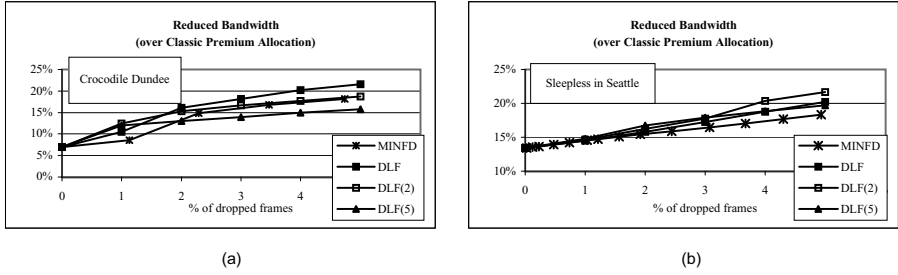
### 5.1 Motion JPEG

**Discard Largest Frames (DLF):** DLF discards the largest  $k$  frames of the video. This algorithm may discard consecutive frames, which may lead to a poor quality playback of the video at the receiver side.

**Discard Largest Frame with distance  $\lambda$  (DLF( $\lambda$ )):** a variation of the previous algorithm. The algorithm uses  $\lambda$  as a parameter that indicates the minimum distance between discarded frames.

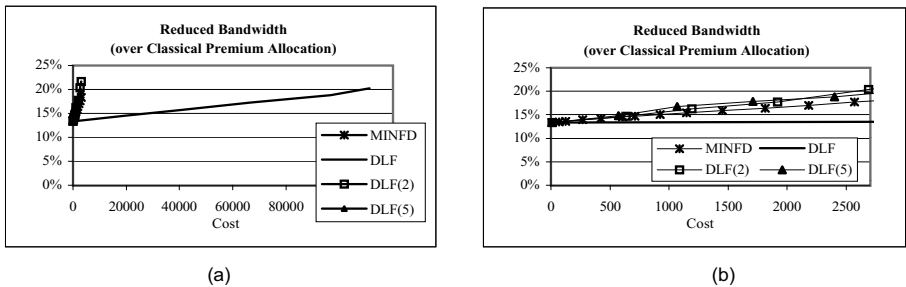
Our experiments compare the heuristic algorithms with the mechanism, described in [5], which uses MINFD to minimize the bandwidth peak rate. We use three algorithms: DLF, DLF(2) and DLF(5) on the MJPEG traces, in order to compute the amount of the reduced bandwidth reached by these algorithms. Fig. 6 illustrates the bandwidth reduction over a peak rate allocation as a function of the number of dropped frames. We observe that our algorithms result in lower bandwidth consumption than MINFD. This is because the mechanism that uses MINFD minimizes the bandwidth peak rate, whereas our heuristic algorithms are concerned with reducing the average allocated bandwidth. In all of the experiments, we observe that discarding a small percentage of frames can greatly reduce the allocated bandwidth.

In Fig. 6(a), discarding 2% of the frames results in a 16% reduction in the allocated bandwidth over a peak rate allocation. All three discard policies perform similarly with DLF slightly better than DLF(2) which is slightly better than DLF(5) as a greater percentage of frames are allowed to be discarded. This is because DLF can discard consecutive frames (in the worst case all of the discarded frames could be consecutive frames and this could result in poor playback quality), whereas DLF(2) and DLF(5) cannot. For *Sleepless in Seattle* (Fig. 6(b)) the three algorithms produce almost the same results, a reduction in the allocated bandwidth from 13% (perfect QoS delivered) to 22% (5% of dropped frames). Readers can refer to [5] for results from analyzing *ET* and *Big*.



**Fig. 6.** (a) *Crocodile Dundee*. (b) *Sleepless in Seattle*.

So far we have focused on the number of dropped frames. Unfortunately there may not be much correlation between this measure of QoS and what the user perceives. Hence we focus on a cost function introduced in [13] which attempts to penalize algorithms that drop neighboring frames. This cost function takes two aspects into consideration: the length of a sequence of consecutive discarded frames and the distance between two adjacent but non-consecutive discarded frames. This cost function assigns a cost  $c_j$  to a discarded frame  $j$  depending on whether it belongs to a sequence of consecutive discarded frames or not. If frame  $j$  belongs to a sequence of consecutive discarded frames, the cost is  $l_j$ , if the frame  $j$  is the  $l_j^{th}$  consecutively discarded frame in the sequence. Otherwise the cost is given by  $1 + 1/\sqrt{d_j}$ , where  $d_j$  represents the distance from the previous discarded frame. More details about this cost function can be found in [13]. We present the cost achieved by the heuristic algorithm when applied to *Sleepless in Seattle* in Fig. 7: DLF performs completely worse than the others, while DLF(2), DLF(5) and MINFD achieve very similar results.



**Fig. 7.** Reduced bandwidth in function of the cost. (b) An enlargement of (a).

Based on this cost function, DLF is not worth using because its cost is too high and because its performance is almost like that of DLF(2) and DLF(5). These two algorithms have very similar values both for the cost and for the

reduced bandwidth. Since DLF(2) results in a greater bandwidth reduction, it is preferred to DLF(5).

## 5.2 MPEG

In the previous section we proposed some heuristic algorithms for MJPEG video. Since the MPEG encoding differs from the MJPEG encoding, we cannot use the previous algorithms for MPEG video. For this reason, in this section, we present some heuristic algorithms specifically designed for MPEG video. In a MPEG video, the frames don't have the same importance as some frames depend on other frames. We use MPEG videos organized in Groups Of Picture (GOP) with a size of 12 frames. MPEG can use three types of frames: *I*, *P* and *B*. The GOP is composed as:  $IB_1B_2P_1B_3B_4P_2B_5B_6P_3B_7B_8$ . To decode a *B* frame, both the previous and future *I* or *P* frames are needed. To decode a *P* frame the previous *P* or *I* frame is needed. An *I* frame can be decoded without the use of any other frames. Thus, the discard of an *I* frame results in the discard of 14 frames (the entire GOP plus the two *B* frames of the previous GOP that depend on the *I* frame), the discard of a  $P_1$  frame results in the discard of 11 frames, the discard of a  $P_2$  frame results in the discard of 8 frames, and the discard of a  $P_3$  frame results in the discard of 5 frames. Only the discard of a *B* frame results in no additional frame discard. Based on these dependencies we propose the following algorithms:

**Drop *I* Frame (DIF):** DIF drops the largest GOPs (plus the two *B* frames preceding the *I* frame) of the video. If  $L$  is the maximum number of GOPs that can be discarded, the algorithm discards the  $L$  largest GOPs of the video. Hence, if  $k$  is the maximum number of frames that can be dropped,  $L = k/14$ .

**DIF( $\lambda$ ):** a variation of the previous algorithm. The algorithm uses  $\lambda$  as a parameter that indicates the minimum distance (in GOP) between discarded GOPs.

**Discard First *P* Frame (DFPF):** DFPF discards the largest groups of frames that depend on the  $P_1$  frame (i.e. 11 frames). Again, if  $L$  is the maximum number of these  $P_1$  group that can be discarded, the algorithm discards the  $L$  largest  $P_1$  groups. In this case  $L = k/11$ .

**Discard Second *P* Frame (DSPF):** DSPF discards the largest group of frames that depend on the  $P_2$  frame. Hence,  $L = k/8$ .

**Discard Third *P* Frame (DTPF):** DTPF discards the largest group of frames that depend on the  $P_3$  frame. Hence,  $L = k/5$ .

**Discard *B* Frame (DBF):** DBF discards only the *B* frames of the video. The algorithm orders the *B* frames and discards the largest  $L$  frames. ( $L = k$ ).

In our experiments we use DIF, DIF(2), DIF(5), DFPF, DSPF, DTPF and DBF in order to compute the amount of reduced bandwidth achieved by these algorithms. Fig. 8 illustrates the bandwidth reduction over a peak rate allocation as a function of dropped frames. We observe that discarding a small percentage of frames can greatly reduce the allocated bandwidth.

In Fig. 8(a) (*The Silence of the Lambs*), DIF and DFPF achieve great results since a drop of 5% of the video results in reducing the bandwidth by 58%. Note that a drop of 1% allows a bandwidth saving of up 42%. In Fig. 8(b) (*Jurassic*

*Park*), a drop of 1% of the video results in a bandwidth reduction of 16%. DIF and DFPP result in a bandwidth saving of 24% when 5% of the video is dropped. With *Starwars* a drop of 1% of the video allows a bandwidth saving of 7% and a drop of 5% of the video results in a bandwidth reduction of 22% (using DIF). With (*MTV*), once again, our heuristic algorithms allow a bandwidth saving of 48% when 5% of the video is dropped and a drop of 1% of the video results in a bandwidth reduction of 43%. For a detailed description readers can refer to [5].

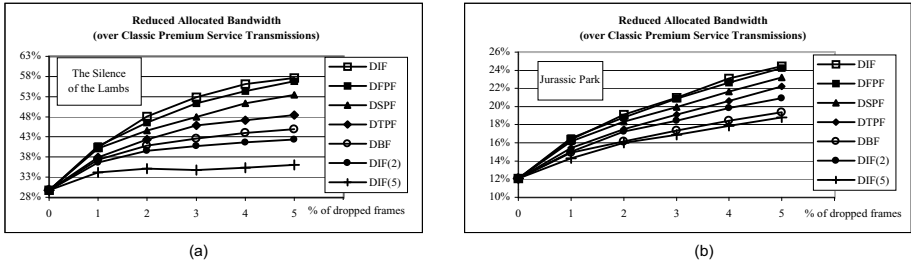


Fig. 8. (a) *The Silence of the Lambs*. (b) *Jurassic Park*.

## 6 Conclusions

In this paper we presented a new BAM that can be used in the premium service architecture introduced in [8] to handle real-time variable bit rate streams over the Internet. We developed it in order to increase the bandwidth utilization for a stream under the premium service architecture. Our BAM is easily implemented in the premium service architecture, as it doesn't require any architectural modification. We show, through several experiments, that its use can greatly reduce the allocated bandwidth for transmitting the same traffic with the same QoS and the same guarantee. The experiments show that our BAM can reduce by up to 30% of the bandwidth needed for transmitting the video.

We showed further benefits possible by sending slightly imperfect QoS video. We developed some heuristic algorithms that can be used to drop frames in order to minimize the bandwidth consumption. All of the experiments presented in this paper show that the use of our algorithms can possibly lead to a great reduction in bandwidth while dropping very few frames. For instance, it is possible to save up to the 43% of the bandwidth dropping only the 1% of the video for *MTV*; up to 42% dropping 1% of *The Silence of the Lambs*.

Since bandwidth is a precious resource, we believe that the proposed mechanisms may be very useful both for the server and the client. The client could be happy to pay less for the same service or for a slightly imperfect service and the server could be happy because reducing the bandwidth needed for one service could mean making bandwidth available for others services.

Our study has assumed knowledge of the bandwidth characteristics of the video stream. This is reasonable in the case of stored video. We are investigating the problem of handling video streams that are generated on-line in order to remove this constraint. Further, we are studying how to modify our BAM in order to introduce interactive controls (like the VCR functions) in the mechanism.

## References

1. Feng, W.C., Sechrest, S.: Critical bandwidth allocation for the delivery of compressed video. *Comp. Comm.* Vol.18, (October 1995) 709-717
2. Feng, W.C., Jahanian, F., Sechrest, S.: Optimal buffering for the delivery of compressed prerecorded video. *Proc. of the IASTED/ISMM*, (January 1995)
3. Feng, W.C.: Video-on-Demand Services: Efficient Transportation and Decompression of Variable Bit Rate Video. Ph.D. Thesis, Univ. of Michigan, (April 1996)
4. Feng, W.C., Rexford, J.: A Comparison of Bandwidth Smoothing Techniques for the Transmission of Prerecorded Compressed Video. *IEEE INFOCOM 1997*, Kobe, Japan, (April 1997) 58-66
5. Furini, M., Towsley, D.: Real-Time Traffic Transmission over the Internet. Department of Computer Science, University of Massachusetts, UMASS CMPSCI Technical Report 99-73 (November 1999)
6. Garret, M., Willinger, W.: Analysis, Modeling and Generation of Self-Similar VBR Video Traffic. *Proceedings ACM SIGCOM*, (August 1994) 269-280
7. Grossglauser, M., Keshav, S., Tse, D.: RCBP: A Simple and Efficient Service for Multiple Time-Scale Traffic. *IEEE/ACM Trans. on Networking*, (1998)
8. Nichols, K., Jacobson, V., Zhang, L.: A Two-bit Differentiated Services Architecture for the Internet. *Internet Draft*, (1998)
9. Salehi, J.D., Zhang, Z.-L., Kurose, J.F., Towsley, D.: Supporting stored video: Reducing rate variability and end-to-end resource requirements through optimal smoothing. *IEEE/ACM Trans. Networking*, vol. 6, (August 1998) 397-410
10. Sikora, J., Teitelbaum, B.: Differentiated Services for Internet2. <http://www.internet2.edu/qos/may98Workshop/html/diffserv.html>
11. Wijesekera, D., Srivastava, J.: Quality of Service (QoS) Metrics for Continuous Media. *Multimedia Tools and Applications*, Vol. 2, No.3, (Sept. 1996) 127-166
12. Wang, Y., Zhu, Q.: Error Control and Concealment for Video Communications: A Review. *Proceedings of the IEEE*, Vol. 86, (May 1998) 974-997
13. Zhang, Z.-L., Nelakuditi, S., Aggarwal, R., Tsang, R.P.: Efficient Selective Frame Discard Algorithms for Stored Video Delivery across Resource Constrained Networks. *Proceedings of INFOCOM 99*, IEEE. New York (March 1999)

# High Speed Wireless Internet Access: Combination of MPLS and BRAN HIPERLAN/2 Technologies

Kallikratidas Nitsos<sup>1</sup>, Tasos Dagiuklas<sup>1</sup>, and Hamed Al-Raweshidy<sup>2</sup>

<sup>1</sup> INTRACOM S.A., Markopoulo Avenue, Peania 190 02, Attika, GREECE  
Tel: (30-1) 6690368, Fax: (30-1) 6860312  
{knit, ntan}@intracom.gr

<sup>2</sup> University of Kent at Canterbury, Canterbury, Kent CT2 7NT, UK  
Tel: (44-1227) 823396, Fax: (44-1227) 456084  
H.Al-Raweshidy@ukc.ac.uk

**Abstract.** This paper discusses the provision of High Speed Wireless Internet Access. It combines the MPLS technology for the core network and the next generation wireless BRAN HIPERLAN2 technology for the access part. The paper focuses on the interworking of MPLS within the HIPERLAN/2, through the design of a Convergence Layer. Two cases have been examined. In the first one, the AP is MPLS capable running a “light” version of the LDP protocol whereas in the second one MPLS is terminated at the edge router. The advantages and disadvantages of both approaches have been presented. Furthermore, the employment of MPLS to provide mobility management functionalities has been addressed. Two mobility management schemes have been examined. In the first one, MPLS is combined with Mobile IP to alleviate the problems of IP-in-IP encapsulation and triangular routing. In the second one, MPLS extensions have been proposed to provide mobility management services in an MPLS Administrative Domain.

## 1 Introduction

Multimedia services require strict QoS requirements in terms of loss and delay. The transmission of these services over IP-based networks, necessitates the forwarding of IP datagrams towards the destination host in a fast and efficient way. The above requirement has led the research community to adopt the Layer 3 switching (IP switching) concept. Several companies (e.g., Nokia IP [1], Cisco [2], Toshiba [3]) have proposed a proprietary IP switching scheme to provide faster routing of the IP datagrams. The IETF established in early 1997 the MPLS Working Group to unify and standardise the various IP switching proposals. The major goal of MPLS is the combination of the benefits of switching and IP routing [4]. On the other hand, the research community has recently paid attention towards the integration of Internet technology and wireless access systems, for the provision of Internet services over wireless/mobile devices. The objective of this paper is the provision of High Speed Broadband Wireless Internet Access by combining the benefits of next generation wireless access networks and MPLS. The driving forces have been the ETSI through the standardisation process for Broadband Radio Access Networks (BRAN- High

PERformance Radio LAN with broadband capabilities – HL/2) and IETF. The innovative aspects of the paper are: the interworking of MPLS within the BRAN HL/2, through the design of a Convergence Layer (CL) and extensions of MPLS to provide Mobility Management by i) combining Mobile IP and MPLS and ii) using the functionality of MPLS.

## 2 MultiProtocol Label Switching (MPLS)

MPLS integrates the label swapping forwarding paradigm with network layer routing. Thus it improves the price/performance of network layer routing, enhances the scalability of the network layer and increases the speed in the Internet [5], [6]. Within the MPLS Administrative Domain, a Label Edge Router (LER) is an MPLS node connecting an MPLS domain with a node outside of the domain. LER is the router that assigns labels and determines explicit routing paths. A Label Switch Router (LSR) is a node, which runs MPLS and is capable of forwarding packets based on labels. Label Distribution Protocol (LDP) is the main signaling protocol used in MPLS for assigning, managing, maintaining and distributing labels.

## 3 BRAN / HIPERLAN 2

ETSI has defined three types of broadband radio networks: HL/2 (25 Mbps typical data rate), HIPERACCESS (25 Mbps typical data rate) and HIPERLINK (up to 155 Mbps data rate). In this paper, we have concentrated in HL/2 systems [7]-[9], because the proposed system architecture includes mobility support functionality. HL/2 provides wireless access to different infrastructure networks (e.g. IP, ATM) by both moving and stationary terminals interacting with access points which, in turn, usually are connected to an IP, ATM or other type of backbone network. For the Data Link Control (DLC) addressing, the following of IDs have been used: a MAC-ID which identifies the terminal; a DLC Connection-ID (DLCC-ID) which identifies the connection within the terminal; an Access Point-ID (AP-ID) which identifies the AP Controller (APC) and AP Transceiver (APT), entities of the AP and a NET-ID which is the same for all APs belonging to the same core network.

## 4 Interworking of BRAN/HIPERLAN2 with MPLS

The interworking between the HL/2 DLC layer and the different network layers defined by the correspondent fora, requires the introduction of a CL between them [10]-[13]. Within the context of this paper, ATM is employed as a transport technology without using the ATM signalling and traffic management functionalities. Thus, the CL must convert the DLC-PDUs into ATM cells, and insert labels in them.

Two different approaches have been followed, i) AP and Edge Router (ER) are MPLS capable, ii) AP is MPLS non-capable and ER is MPLS capable.

The term “MPLS capable” as applied to the AP means that through the employment of labels, LDP does not establish paths but inserts labels in order to pass BRAN-related information to the ER. Therefore, a “light version” of LDP runs on the AP. The operation of LDP on the ER remains the same, since Fixed Terminals (FTs) may be connected to the ER.

4.1 Both AP and ER Are MPLS Capable

Upstream means that data are transmitted from the MT towards the core network, while downstream is the opposite direction.

**Upstream.** The IP layer in the MT is integrated over the BRAN DLC layer through the CL. Peer-to-peer communication, between the AP and the MT takes place and in turn the AP communicates with the ER. Hence, the ER knows the IP addresses of the MTs (stores them in a table) attached (via the AP) to it.

The MT receives from the AP Information Elements (IE), containing the MAC-ID and the DLCC-ID that the AP assigns to the MT. The MT sends a DLC-PDU to the AP as defined by BRAN. On receipt of the DLC-PDU, the AP strips the FU, the SN and the CRC bits, to construct the DLC-SDU. It will read the CL-Tag field of the received DLC-PDU and will understand to which CL the frame must be sent. The rest of the process is shown in Fig.1-a. The CPCS-SDU will be sent to the appropriate Service Specific Convergence Sublayer (SSCS) according to the type of higher layer. The IP datagram will be formed which will be segmented (Fig.1-b) to the appropriate frame type (e.g. Frame Relay, Ethernet, ATM etc.), and then will be sent towards the Edge Router.

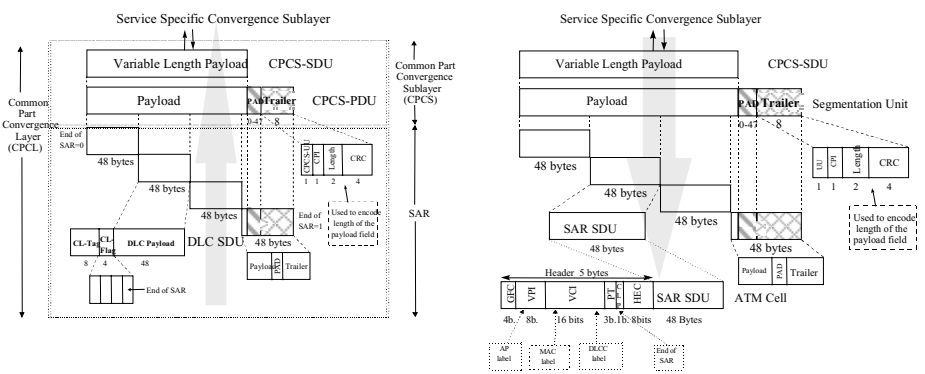


Fig. 1. (a) Reassembly of DLC-PDUs at the AP; (b) Segmentation to form ATM cell, at the AP

However, the VPI/VCI field of the ATM cell will not convey standard ATM information. Since the AP knows that the received DLC-PDU comes from a certain MT, it knows the MAC-ID and the DLCC-ID of the sender. Furthermore, it knows the AP-ID of itself. This information must be mapped into labels, which is inserted in the VPI/VCI fields. The total length of MAC-ID, AP-ID and DLCC-ID is 22 bits, which can be conveyed in the VPI/VCI field (24 bits long). The “End of SAR” information,



which is retrieved from the last bit of the CL-Flag field, will be inserted in the PT field (see Fig.1-b). The AP knows the MAC-ID of the MT that sends the frames and the ER knows the IP address of this MT. Therefore, they must agree on a label binding to associate this information.

In cases where ATM is used as a layer 2 technology then the re-assembly of DLC-SDUs, to form the CPCS-SDU, and re-segmentation to form the ATM cells, is avoided, since the payload in both DLC-PDUs and ATM cells is 48 bytes long. A change in the header, simplifies the entire procedure, i.e. strip the CL-Flag and CL-Tag fields and insert the ATM header.

The ATM cells can then be transmitted to the ER. The ER receives these ATM cells which are then reassembled to form the IP datagram. The IP datagram is segmented through AAL5 to ATM cells and the ER assigns in the VPI/VCI field the 20-bit label (possibly establishes the LDP path for explicit routing, too). The existing LDP algorithms, as defined by IETF, run at the ER. The ER must have a table to map the received labels from the AP to IP addresses. It must then assign packets to particular Forwarding Equivalent Classes (FEC), map each FEC to a Next Hop Label Forwarding Entry (NHLFE) and finally map each one or a set of NHLFEs to a label. This information will be stored in a Forwarding Information Base (FIB).

The "labelled" ATM cells are now delivered to the core network and according to the labels distributed by LDP, layer 3 switching is performed up to the ER connected to the destined MT.

When a FT or a MT sends a packet to another MT, then the packet will be forwarded according to the IP address. The ingress MPLS router will insert a label according to the destination IP address. The labelled packet will reach the ER which holds all IP addresses of the MTs attached to it. It will assign a label to forward the frame to the AP. Since the AP knows that this label corresponds to the MAC-ID, the frame reaches at its destination.

**Downstream.** The ER receives labelled ATM cells. The MPLS label is included in the VPI/VCI field and contains information related to the destination and source address (the ER has a table with mappings of MT IP-address to DLCC-label, MAC-label). After passing through AAL5, the cells are reassembled and the IP datagram is constructed. The above information is stored and again through AAL5, ATM cells are formed containing additional information. The VPI/VCI field contains an End of SAR, a DLCC label, a MAC label and a AP label according to the IP addresses, respectively (as illustrated in Fig.1-b).

At the AP, the 5-byte header of the received ATM cells is stripped and the remainder (SAR SDU) is reassembled to form the Segmentation Unit and the CPCS-SDU. According to the labels in the VPI/VCI field, the AP has a table, which maps them to MAC-IDs and DLCC-IDs. This way the AP knows the MTs where the DLC-PDUs must be forwarded.

The interworking function is responsible for segmenting the CPCS-SDUs to DLC-PDUs and mapping the DLCC labels and MAC labels to MAC-IDs and DLCC-IDs. Finally, the DLC-PDUs are transmitted via the air interface to the MT.

The MT receives them and extracts the FU, SN and CRC fields to form the DLC-SDUs. From the last bit of the CL-Flag, the end of a message is identified.

## 4.2 AP MPLS Non-capable and ER MPLS Capable

**Upstream.** The procedure is the same as described in section 4.1. In this scenario, it is not required to send information to the ER, related to the wireless access part, since the source IP address is known to the ER. Therefore, the ATM cells sent from the AP to the ER contain no labels. The ER receives the ATM cells and through the SAR sublayer, the IP datagram is formed. The ER reads the destination IP address, assigns the packet into a FEC and assigns a label according to that FEC (acts as an ingress router in an MPLS cloud). The MPLS label is encapsulated in the VPI/VCI field of the ATM cell and forwarding is implemented as defined by the IETF's MPLS Working Group.

**Downstream.** The ER acts as an egress router to an MPLS domain. It reads the label (in the VPI/VCI field of the ATM cells) and maps it to a FEC, which in turn is mapped to an IP address. The ATM cells are reassembled to form the IP datagram. The destination IP address is stored. The ER will have a table to map MT IP addresses to AP IP addresses where the MT is associated with. For this purpose an ATM VP is established from the ER to the AP. The IP datagram is segmented into ATM cells through AAL5 SAR sublayer and will leave the ER destined to the AP.

At the AP, the ATM cells are reassembled, forming the IP datagram and according to the destination address will be forwarded to the appropriate MT, after segmenting the datagram to DLC-PDUs. If there is no connection established with that MT (i.e., AP has not assigned a DLCC-ID to it), a DLCC-ID is assigned and the packet is forwarded to the MT. The MT receives the DLC-PDUs and extracts the FU, SN and CRC fields to form the DLC-SDUs. The last bit of the CL-Flag is used, in order to recognise the last DLC-SDU of the message.

## 4.3 Comparison of Two Approaches

The approach discussed in section 4.1, is advantageous since there is no reassembly of IP datagrams at the AP. This is the reason why the AP maps the MAC-ID to a label and not directly to an IP address. The label will be mapped to an IP address at the ER. Thus, traffic destined for the MT is routed up to the ER according to the IP address (or labels if it is an MPLS Domain connected to the ER), up to the AP according to the labels (of the "light version" LDP) and up to the MT according to the DLCC-ID and the MAC-ID.

Additionally, if the AP is MPLS capable, communication between two MTs attached under the same AP can be realised without involving the ER. This way, the communication between the MTs is faster.

Finally, the second approach has the disadvantage of reassembling the IP datagram at the AP.

# 5 Mobility Management in MPLS Domain

There is a wide range of technologies for supporting mobile users (e.g. GSM, PCS, UMTS, Mobile IP [14]-[16] and Wireless ATM [17],[18]). All these technologies support two fundamental mechanisms:

- Location management -locating users prior to or during connection establishment
- Handover (HO) -rerouting connections when users move.

Mobility management within IP-based networks could be performed using Mobile IP. However, Mobile IP has certain disadvantages. MPLS could alleviate these as will be described in section 5.1. Moreover, further study has been undertaken to provide Mobility Management using a “Mobility Server” approach within an MPLS Domain (section 5.2). In this case, MPLS signaling protocols (LDP) and MPLS messages are employed to perform mobility management functionalities.

## 5.1 Use of Mobile IP

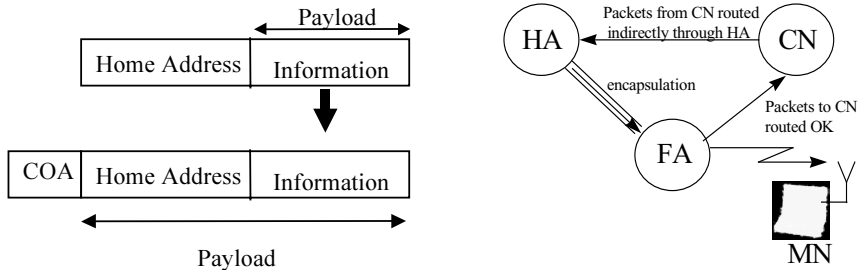
Current Internet Protocol versions do not support host mobility. IP has been designed such that moving hosts are not considered. Thus, the rise of Mobile IP: an Internet protocol designed to support host mobility [14]-[16]. Mobile IP is consisting of the following entities: Mobile Node (MN), Home Agent (HA), Foreign Agent (FA) and Correspondent Node (CN) and makes use of two addresses, a Care-of-address (COA) and a Home Address.

When a CN sends a packet to a MN, if the MN is on its home network the packet will reach the HA and will then be delivered to the MN. If the MN has changed its point of attachment, then the packet will reach the HA. The HA is informed of the new point of attachment of the MN (through the FA), and will forward the packet to the FA that the MN is currently attached. The HA uses an encapsulation process to forward the packet to the FA. It encloses an IP datagram within another IP header, which contains the care-of address of the mobile node. The IP datagram itself remains intact and untouched throughout the enclosing process (Fig.2-a).

**Problems with Base Mobile IP Protocol.** The problem with this approach is the encapsulation process [19]. When the MN is located at the foreign network an additional IP header (24 bytes) is inserted to the IP packet. This is too much overhead to the final datagram (several fields are duplicated from the inner IP header). This waste of bandwidth is uneconomical. IETF has defined a so-called Minimal Encapsulation scheme, [20], as an alternative to encapsulate the datagram. The approach to the encapsulation method is as follows: Instead of inserting a new header, the original header is modified to reflect the COA. A minimal forwarding header is inserted in between the modified IP header and the unmodified IP payload, to store the original source and destination address. When the FA tries to decapsulate the received IP datagram, it will simply restore the fields in the forwarding header to the IP header, and then remove it. There is a restriction to the use of this encapsulation method. If the original datagram is already fragmented, then minimal encapsulation must not be used since there is no room left to store fragmentation information.

One other problem of Mobile IP regards “triangle routing”. Suppose that the CN wants to send a datagram to the MN (Fig.2-b). Based on the base Mobile IP protocol, the CN will send the datagram to the MN’s HA, which may be a half globe away. The HA will then forward the datagram to its COA, which might just take a half second to reach if the datagram is sent directly from the CN. This kind of “indirect routing” is inefficient and undesirable. This problem, can be alleviated through a route optimisation procedure as defined by the Mobile IP Working Group of the IETF [21].

The key approach to route optimisation is as follows: Binding cache, containing the mobility binding of MNs, is provided for the CN that looks for optimising its own communication with MNs. In this way, the CN has a way to keep track of where the MNs are. So when the CN wishes to send the datagram to the MN, the IP datagrams are sent directly to the destination address, eliminating the “zig-zag” routing. The means for the MN’s previous FA to be notified of the MN’s new location is provided. This mechanism allows datagrams in flight to the MN’s previous FA to be re-directed to its current address.



**Fig. 2.** Problems with base Mobile IP: (a) IP encapsulation; (b) Triangle routing

**Solutions Using MPLS.** These problems can be alleviated through the employment of MPLS. By making the HA, the FA, the CN and the MN MPLS capable, we propose the following solutions:

- The “IP-within-IP” encapsulation method can be replaced by the use of label stack offered by MPLS. Forwarding will be based on MPLS labels. Thus, instead of inserting a new IP header (24 bytes long), a level 2 label can be used that will be pushed on the label stack by the HA whenever the MN has changed its point of attachment. Therefore, the overhead introduced with MPLS is smaller (MPLS header is 32 bits long) than the overhead introduced by the IP-within-IP encapsulation.
- The “triangle routing” problem can be alleviated according to the following procedure: Suppose that the CN wants to send a packet to the MN, which is attached to a FA. The CN will send a “label request” message to the HA, using a level 1 label. The HA will forward the “label request” message to the FA by pushing a new level 2 label on the label stack. Since the labelled packet received by the FA arrives only with the level 1 label (the penultimate hop of the FA pops out the level 2 label), the path from the HA to the FA is transparent to the FA. Hence, the FA will send back a “label mapping” message directly to the CN. As a result, an LDP path is established between the CN and the MN. The CN sends the packet with the necessary information towards the MN. This way “triangle routing” is only followed by the “label request” message, before the beginning of communication between the CN and the MN. Furthermore, it is only followed once, prior to the LDP path establishment, and when data is sent, the LDP path is followed, without involving the HA (Fig.3).

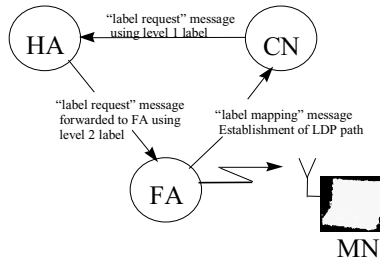


Fig. 3. Alleviation of “triangle routing” using MPLS

## 5.2 Using a Mobility Server within an MPLS Domain

In this approach Mobility Management functions are provided using MPLS signalling protocols (LDP) and MPLS specific messages.

**Location Management.** Each MPLS Administrative Domain contains a Mobility Server which comprises a Home and a Visitor Database. Both Databases have a table for mapping MT IP addresses to NET-IDs (representing identifier of each MPLS Administrative Domain). The Home Database has entries from MTs belonging to the current MPLS Administrative Domain. The Visitor Database has entries from visiting MTs in the current MPLS Administrative Domain.

Each time a MT powers on, the Mobility Server checks its Home Database. If there is an entry of the MT’s IP address it assumes that the MT is located the home MPLS Administrative Domain. Otherwise, the Mobility Server tries to find the MT’s “home server”, to learn information for the MT and perform authentication. This information for reaching the “home server” will be retrieved by the MT’s NET-ID.

All Mobility Servers can be members of a Virtual Private Network (VPN). MPLS provides a simple and efficient solution for VPNs, offering a viable VPN service to the members. This way all Mobility Servers will communicate efficiently, rapidly and with security. Among the Mobility Servers, Constraint Based LSPs will be established. After the Mobility Server has authenticated the MT and received all information from the MT’s “home server”, it will add an entry in the Visitor Database with the MT’s IP address mapped to a NET-ID.

**Handover.** One of the functional entities of the DLC layer defined by ETSI is Radio link Control Protocol (RCP) [22]. Within RCP, the Radio Resource Control (RRC) entity, is responsible for the HO. There are three different types of HO; Sector HO, Radio HO (intra-AP) and Network HO (inter-AP) which involves the CL and Higher Layers. In the last case, specific signalling is required via the core network. Since the AP is changed, the new and the old APs must communicate to exchange information. Three different types of Network HO have been examined: a) Inter-AP, Inter-ER (Edge Router), b) Inter-AP, Intra-ER and c) Inter-ER, Inter-Domain.

*Inter-AP, Inter-ER HO.* This case is illustrated in Fig.4-a. When the MT detects the need for a HO to another AP, it may be still synchronised to the current AP. In this case, the MT may notify the current AP that it will perform a HO to another AP. The

notified AP shall stop transmitting/receiving data to/from this MT, but shall maintain association for a specific time, when indicated.

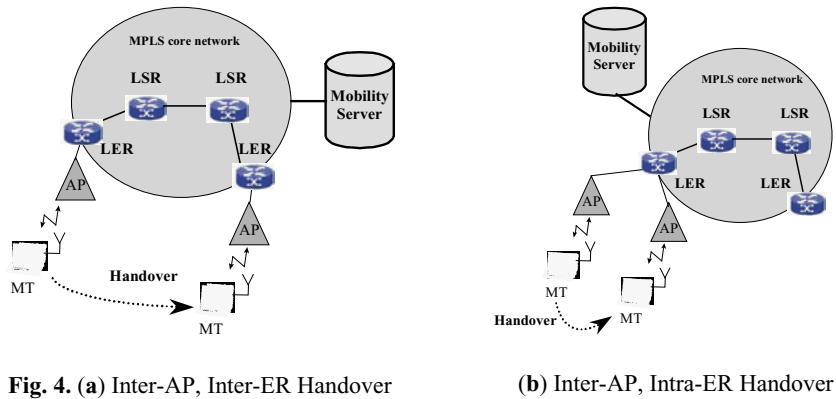


Fig. 4. (a) Inter-AP, Inter-ER Handover

(b) Inter-AP, Intra-ER Handover

Fig.5 illustrates the inter-AP, inter-ER MSC. In the *“Handover\_Request”* message, security parameters such as MAC-ID, NET-ID and AP-ID of the previously connected AP must be indicated. The old AP will be contacted via the backbone network, through the use of MPLS labels. All the parameters related to the on-going connections and security functions must be available in the new AP in order to maintain the association state. These will be sent by the old AP to the old ER and from the old ER to the new one through an LSP tunnel. The new AP negotiates the received parameters and it updates its routing tables about the new point of attachment of the MT.

All traffic sent to the old AP destined for the MT during HO, will be sent to the new AP through the LSP tunnel. This will be kept alive until the explicit LSP path has been established from the new ER towards the CN. The explicit LSP path is established from the new ER by sending a *“Label\_Request”* message to the CN. The CN replies with a *“Label\_Mapping”* message. The explicit LSP path is established and all connections are redirected. The old AP can now perform disassociation of connections with the MT. The *LDP\_Handover\_Alive* and *LDP\_Handover\_Reply* messages will be sent as LDP specific messages following an explicit or hop-by-hop tunnel. These type of messages are not defined by the MPLS WG, therefore they must be defined. The “Type” range 0x3F00 through 0x3FFF is left by the MPLS WG for extensibility. Following the LDP specification, the encoding scheme used by LDP is Type Length Value (TLV) encoding.

The structure of the *“LDP Handover Alive”* and *“LDP Handover Reply”* messages (Fig.6), follow the structure defined by IETF [23]. In addition, it uses Type-Length-Value (TLV) encoding scheme to encode the information carried in LDP messages.

*Inter-AP, Intra-ER Handover.* For the case where the MT changes AP but the new AP is under the same ER (Fig.4-b), the MSC is illustrated in Fig.7-a.

*Inter-ER, Inter-Domain Handover.* This scenario is illustrated in Fig.7-b. The new LER tries to find the old one to retrieve relevant information. LERs hold NET-IDs (mapped to IP addresses) of MTs belonging in their MPLS Domain. When the MT has changed Domain it will not find an entry of the NET-ID. Therefore, the LER

requests from the Mobility Server the NET-ID-IP address binding. The Mobility Server will contact (through the explicit paths between the two members of the VPN) the Mobility Server of the MPLS-Domain where the MT was previously attached. The Mobility Server will then inform the new LER of the IP address of the old LER. Now the new LER can contact the old one as described previously. Therefore, the role of the Mobility Server is to provide the IP address of the LERs that do not belong to the same MPLS-Domain. This way, LERs will have much fewer entries in their tables. They will have entries (NET-ID - IP address binding) only of LERs that belong to the same domain.

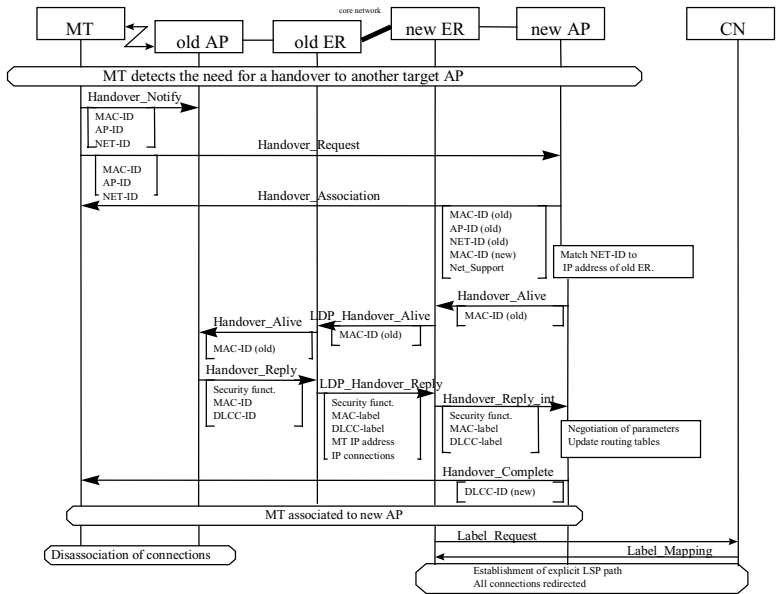


Fig. 5. Proposed Message Sequence Chart for the Inter-AP, Inter-ER case

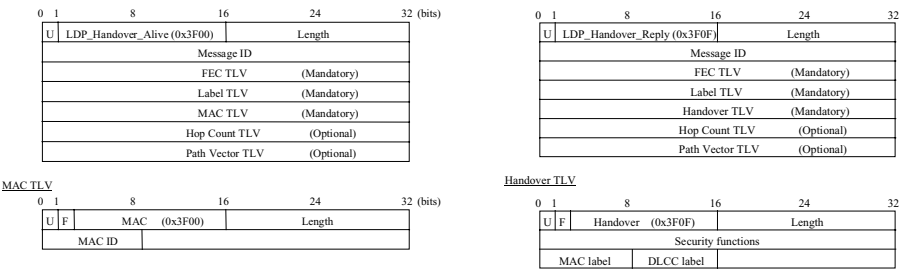
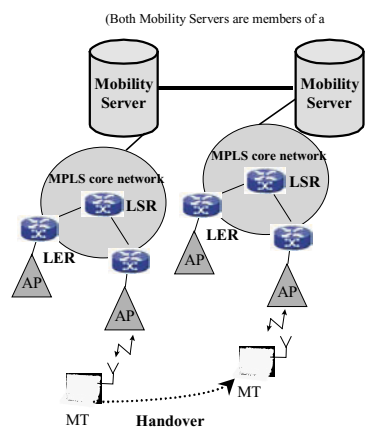
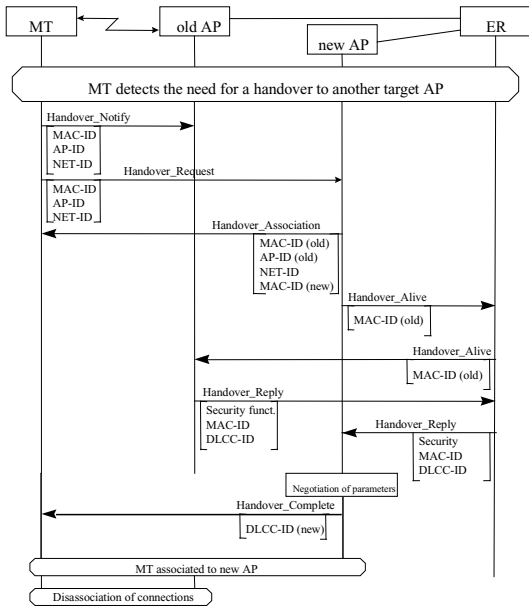


Fig. 6. (a) "LDP Handover Alive"

(b) "LDP Handover Reply"



**Fig. 7. (a)** Proposed Message Sequence Chart for the Inter-AP, Intra-ER case **(b)**Inter-ER, Inter-Domain Handover

**5.3 Comparison of Two Schemes for Mobility Management**

The first difference between the two approaches is that with Mobile IP there are two addresses: the Home Address and the Care of Address. In the Mobility Server approach there is only one address (like in the HLR/VLR approach of GSM).

Furthermore, in the Mobility Server approach, the “visitor” Mobility Server communicates, via the VPN, with the “home” Mobility Server. After receiving all MT-related information, it updates its database and acts as a “home server”. However, with the Mobile IP approach, the HA is informed of the new location of the MT and forwards traffic to the FA which will deliver it to the MT, creating the “triangle routing” problem. Additionally, with route optimisation, the CN is informed of the new location and not the HA. Hence, it directly sends the packets to the MT, without involving the HA.

**6 Conclusion**

This paper describes how two technologies (MPLS and HL/2) can interwork in order to provide High Speed Wireless Internet access. Two scenarios have been examined. In the first one, the AP is MPLS capable whereas in the second scenario MPLS is terminated at the LER.



Furthermore, MPLS is employed to provide mobility management. Two schemes have been examined. In the first one, MPLS is combined with Mobile IP to alleviate the problems of IP encapsulation and triangular routing. In the second one, MPLS extensions have been proposed in order to provide mobility management services in an MPLS Administrative Domain through the employment of a Mobility Server. A more detailed analysis of this concept can be found in [24].

## References

1. Ipsilon, "General Switch Management Protocol Spec. v2.0", *RFC 2297*, March 1998.
2. Rekhter D. et al, "Tag Switching Architecture-Overview", *RFC-2105*, February 1997.
3. Toshiba Press Release, "Toshiba and Cisco to Cooperate for Next-Generation High Speed Interworking Technology", <http://www.toshiba.co.jp/about/press>, 11 November 1996.
4. Callon R. et al, "A Framework for MPLS", *Internet Draft*, November 1997.
5. Rosen E. et al, "MPLS Architecture", *Internet Draft*, February 1999.
6. ACTS ITHACI, "IP Switching: Current Technologies and Future Requirements", *Deliverable*, July 1998.
7. ETSI BRAN, HIPERLAN/2/HIPERLINK, "Requirements and architectures for Wireless Broadband Access and Interconnection", *DTR-BRAN-010002v0.1.2*, July 1998.
8. ETSI BRAN, "Wireless Access to the Information Age", 1998.
9. ETSI BRAN, HL/2, "System Overview", *BRAN\_ETR\_030002v0.2.0*, April 1999.
10. Telia Research, Ericsson, "Outline of the IP Convergence Layer TS", *BRAN contribution, HL12.5TRSIa*, 15 February 1999.
11. ETSI BRAN, "IP Convergence Layer for HIPERLAN 2 and HIPERACCESS", *DTS/BRAN-0024004 v0.0.1*, April 1999.
12. Ericsson, "Convergence Layer Basics", *BRAN contribution, 3ERI104A*, May 1999.
13. ETSI BRAN, "Higher Layer Information IP CL", *HL12ERI4A*, Ericsson, May 1999.
14. Perkins C. et al, "IP Mobility Support", *IETF RFC 2002*, October 1996.
15. Perkins C., "Mobile IP: Design Principles and Practices", *Addison Wesley*, 1998.
16. Becker C. et al, "IP Mobility Architecture Framework", *Internet Draft*, September 1999.
17. Ayanoglu E. et al, "Wireless ATM: Limits, Challenges and Proposals", *IEEE Personal Communications Magazine*, August 1996.
18. Acharya A. et al, "Mobility Management in Wireless ATM Networks"
19. Perkins C., "IP Encapsulation within IP", *RFC 2003*, 6 July 1995.
20. Perkins C., "Minimal Encapsulation within IP", *RFC 2004*, 6 July 1995.
21. Perkins C., "Route Optimization in Mobile IP", *Internet Draft*, 25 February 1999.
22. ETSI BRAN, HIPERLAN2; Functional Specification, "Radio Link Control Protocol (RCP) Service Description", *Version 2, Revision2*, May 1999.
23. Andersson L. et al, "LDP Specification", *Internet Draft*, May 1999.
24. Nitsos K., "High Speed Wireless Internet Access: Combination of MPLS and Next Generation Wireless Access Networks (BRAN-HIPERLAN/2)", MSc. Dissertation, September 1999.

# Effect of Turn-Around Times on the Performance of High Speed Ad-hoc MAC Protocols

Ajay Chandra V. Gummalla and John O. Limb

Georgia Institute of Technology,  
Atlanta GA 30332, USA

**Abstract.** Many random access protocols have been proposed for ad-hoc networks and they are based on collision avoidance principles and are designed for low data rates ( $\leq 2Mbps$ ). A key element in the design of high speed MAC protocols is the ability to send feedback from the destination to the source about the state of current transmission. We present an analysis to show that when hardware constraints like transceiver turn-around times are involved, multiplexing the feedback using a different frequency channel is a better design choice. We validate this analysis by comparing the performance of three current ad-hoc MAC protocols as the data rates are increased using a power law model for the variation of the turn-around time. We show that at high data rates, the busy tone protocols will be much more efficient than the collision avoidance protocols. Further, the results show that unless the turnaround times scale in proportion to data rate the performance of CSMA/CA protocols will be worse than slotted-ALOHA

## 1 Introduction

Technological advances coupled with the flexibility and mobility of wireless systems are the driving force behind *Anyone, Anywhere, Anytime* paradigm in wireless networking. Wireless networks that can be rapidly deployed and those that do not need any pre-existing communication infrastructure play an important role in enabling this paradigm. Such network architectures, called ad-hoc networks, are a topic of extensive research. Medium access control (MAC) protocols define rules to allow efficient and fair access to the shared wireless medium and thereby play a crucial rule in determining the performance of such networks. The MAC protocols for such networks should be distributed random access protocols because of their architecture that has no infrastructure.

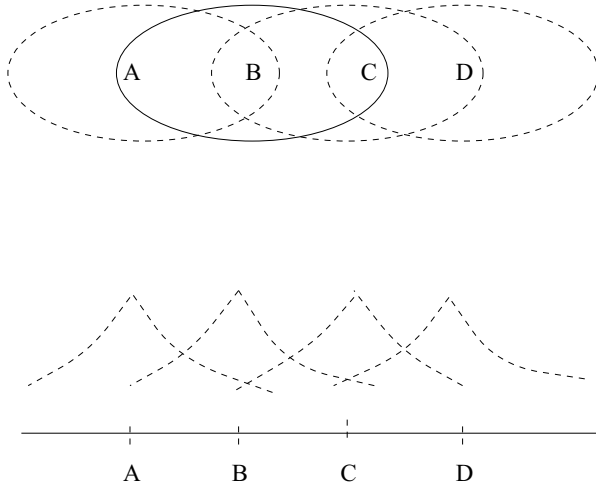
Wireless random access protocols have been extensively researched and a plethora of protocols proposed. The nature of the wireless channel (time-varying channel; location dependent carrier sensing) and the constraints of the wireless transceivers (half duplex mode of operation; transceiver turnaround times) make the design of ad-hoc wireless MAC protocols very challenging. The scalability of the protocol with data rate adds another dimension to the problem. It is well understood that the design of a MAC protocol to handle these constraints requires

feedback from the destination node (DN) to the source node (SN) (section 2). There are two options for multiplexing this feedback. One is to use different time slots which we refer to as time duplexed feedback. The other is to use different frequency bands for the data channel and the feedback channel. This is referred to as frequency duplexed feedback. Based on how the feedback is multiplexed all the ad-hoc MAC protocols proposed can be classified into two categories namely; handshaking protocols which use time-duplexed feedback, and busy-tone protocols which use frequency duplexed feedback.

Given the hardware constraints like turn-around times which is the better choice? We present an analysis which shows that frequency duplexed feedback is a better option (section 3). To validate the analysis, we study the performance of the two CSMA/CA based protocols and one busy-tone protocol as data rates are increased, assuming that the turn-around times decrease according to a power law (section 4). We see that as data rates are increased, CSMA/CA based protocols are very inefficient. Further, even at the data rates at which these protocols have been proposed, the efficiency of these protocols is poor when transmitting small packets (section 5).

## 2 Motivation

Carrier sensing wireless protocols have to accommodate many wireless channel characteristics like burst errors, an unreliable channel, location dependent carrier sensing and hardware constraints like turn-around times. The efficiency of a random access protocol is determined by how fast collisions are detected and how soon this information can be conveyed to the source node. In wireline protocols like CSMA/CD, the ability of a node to listen to the medium while transmitting and the fact that a collision on the medium is heard by all nodes listening to the medium results in high performance. In the wireless medium both the above assumptions break down. First, a wireless transceiver cannot be transmitting and listening to the medium at the same time because of self interference (Any transmitted signal that leaks into the receiver usually has a much higher energy than the received signal and hence transceivers cannot listen and transmit at the same time). In the wireless medium the signal strength falls off as a function of distance. Hence, depending on the position of a node relative to the source node, channel sensing will produce different results. This is known as location dependent carrier sensing. Consider the scenario in Fig.1 where B is in radio coverage of A and C and C is in range of B and D. If B is transmitting, A and C sense the channel as busy while D thinks the channel is idle. A transmission from D will corrupt data reception at C. Therefore, location dependent carrier sensing is a serious problem and results in unfair sharing of bandwidth and poor efficiency. Unlike in a wired media, two simultaneous transmissions do not imply a collision. A collision occurs only if the destination node (DN) cannot decode a transmission. Therefore, the DN is the only node that can identify a collision. When a collision is detected, this information should be conveyed to the source node (SN), so that it can abort its transmission and minimize wasted capacity.



**Fig. 1.** Logical abstraction of an ad-hoc network.

The feedback problem is: how to enable the destination to convey information about the state of transmission (idle/collision) to the source node? A feedback channel is not available in current wireless systems. Therefore protocols try to minimize collisions by exchanging handshaking messages to reserve the channel for data transmission. This handshaking can be considered as time duplexing of feedback information. Due to the half duplex operation, wireless transceivers would need to switch from receive mode (listening to the channel) to transmit mode (sending data) and vice versa. During such switching there are time periods when the transceiver can neither receive or transmit data. Such switching times also known as turn-around times should be considered in the design of protocols. Consider a turnaround time of  $10\mu\text{s}$ . This corresponds to an overhead of 10bits at 1Mbps and an overhead of 1000bits (125bytes) at 100Mbps. An RTS-CTS-DATA-ACK handshake in the IEEE802.11 MAC protocol requires four turnaround times [6]. The second alternative is to use a feedback channel at a different frequency which has a higher hardware cost because of the need for two transceivers; one each for the data channel and the busy-tone channel. The challenge, then, is: considering the physical channel and hardware limitations which is the most cost-effective method of duplexing the feedback channel?

### 3 The Duplexing Problem

#### 3.1 Assumptions

The following assumptions are made about the channel and its capacity. Consider a wireless channel that has bandwidth  $B$ . This channel is used for communication among multiple users who share the same physical medium. From the channel noise and other constraints, independent of MAC protocol, a maximum data

rate of  $R$  bps can be achieved on this channel. However, when multiple users communicate with each other without any kind of co-ordination in a distributed manner, there is a possibility that multiple users transmit at the same time and such transmissions result in bandwidth being wasted.

Definitions in the analysis (These parameters are illustrated in Fig.2):

Packet Length =  $L$  bits

Overhead when a collision occurs =  $O_C$  bits

Overhead when a successful transmission occurs =  $O_S$  bits

Overhead when the channel is idle =  $O_I$  bits

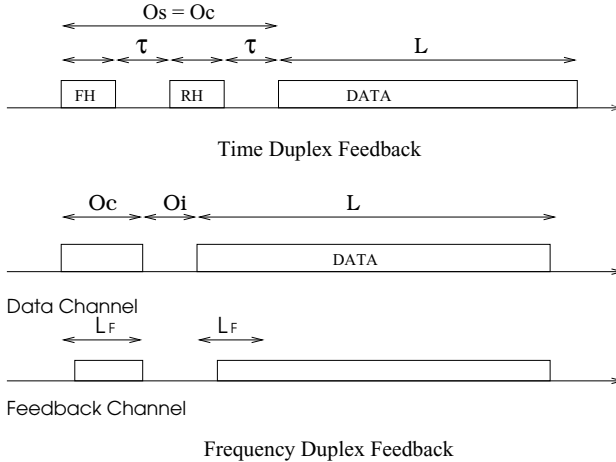
Probability of a successful transmission given a transmission attempt =  $P_S$

Probability of channel being idle =  $P_I$

Using the renewal period concept the efficiency is given by [8],

$$\eta_{mac} = \frac{P_S L}{P_S(L + O_S) + O_I P_I + (1 - P_S)O_C}$$

In this equation, the two probabilities  $P_S$  and  $P_I$  are a function of the load on the network and the contention resolution algorithms. These probabilities have little effect on the efficiency when the backoff parameters are chosen appropriately. The actual efficiency is controlled by the overheads ( $O_C$ ,  $O_I$  and  $O_S$ ) in relation to the packet size. Therefore in this analysis, we take one *typical* handshaking protocol and one *typical* frequency-duplexed feedback protocol, estimate the corresponding overheads and determine their approximate efficiency.



**Fig. 2.** Description of different overhead components in the protocols. FH and RH are the forward and reverse handshaking messages.  $\tau$  is the turnaround time.

### 3.2 Time Duplexed Feedback Channel (Handshaking Protocols)

A typical handshaking protocol works as follows: After sensing the channel for a period of time, the source initiates a transmission attempt with a forward handshake (FH). The destination responds with a reverse handshake (RH) if the FH was successful. A successful FH-RH reserves the channel and data is transmitted. A collision can occur on the FH or RH and the resulting overheads are slightly different. However, to understand the influence of these turnaround times let us make the simplifying assumption that the overhead of both types of collision is the same and it is equal to the overhead in a successful transmission. The time spent sensing the channel idle is typically much smaller than the overhead due to collision and successful transmission. Therefore we neglect the overhead of the idle period. Lets consider the overhead in transmitting a single handshake message. A node has to first sense the channel till it finds an idle slot. Once an idle slot is found, the node has to switch from listen mode to transmit mode (1 turnaround period). The node then transmits the handshake packet. This handshake packet has to be large enough such that the destination station can detect the packet transmission ( $L_{BD}$  symbols), acquire the burst ( $L_{BA}$  symbols) and then understand the MAC address information ( $L_M$  symbols). Further this packet size has to be greater than a roundtrip delay in the network so that every node can receive this packet unambiguously. Let the number of turnarounds in one successful transmission be  $k$  ( $=3$  in Fig. 2) and each turnaround time is  $\tau$  secs long. The overhead due to successful transmission or collision is given by,

$$O = O_S = O_C = k(\tau R + L_{BA} + L_{BD} + L_M)$$

and the efficiency is given by,

$$\eta_{mac} = \frac{L}{L + \frac{O}{P_S}} = \frac{1}{1 + k(\frac{L_{BA} + L_{BD} + L_M + \tau R}{LP_S})}$$

The efficiency of the MAC protocol has an inverse relationship to the data rate if  $\tau$  does not scale in proportion to the data rate. Given a fixed turn-around time, the handshaking protocols do not scale well with data rate. Further, each burst transmission has a burst header ( $L_{BA} + L_{BD} + L_M$ ) which can become a significant overhead when transmitting small packets (e.g. a TCP ACK message is 40 bytes and it comprises 86% of WWW traffic [9]).

### 3.3 Frequency Duplexed Feedback Channel (Busy-Tone Protocols)

In a frequency duplexed feedback channel a typical packet transaction is as follows: the source after sensing the medium idle will initiate the data transmission. The destination sends the information about success or collision back to the source on the feedback channel. The source node does not know the result of its current transmission till it gets this feedback. The available bandwidth has to be allocated between the data channel and the feedback channel. If we allocate this bandwidth in ratio  $\alpha$ , the data and feedback channel capacities are related

as  $B_F = \alpha B_D$  and  $R_F = \alpha R_D$ . We can assume that it takes the same time to detect the transmission, acquire the burst and detect the source and destination of the transmission, as in a handshaking protocol. The time to get the feedback to the source is therefore,  $\frac{L_{BA}+L_{BD}+L_M}{R_D} + \tau_a$  where  $\tau_a$  is the time required to assert and detect the feedback tone. The time required to detect the feedback tone should equal  $\frac{L_{BD}}{R_F}$  (the same time as needed to detect the data carrier, but scaled by the feedback channel bandwidth). Therefore, the overhead in bits is  $O = L_M + L_{BS} + L_{BA} \frac{1+\alpha}{\alpha}$ . The efficiency is given by

$$\eta_{mac} = \left( \frac{1}{1+\alpha} \right) \left( \frac{L}{L + \frac{O}{P_s}} \right) = \left( \frac{1}{1+\alpha} \right) \left( \frac{1}{1 + \frac{L_M + L_{BS} + L_{BA} \frac{1+\alpha}{\alpha}}{LP_s}} \right)$$

It can be seen that the efficiency of frequency duplex solutions is independent of the data rate. Also the efficiency is a function of the ratio parameter  $\alpha$ . For small values of  $\alpha$  it takes too long to receive the feedback information resulting in poor efficiency. At the other extreme, if one assigns too much bandwidth to the feedback channel, the time to transmit data increases, thereby decreasing the efficiency. Therefore there is an optimal choice for  $\alpha$ . The efficiency is maximized when  $\alpha = \left( \frac{L_{BD}}{L_{BD}+L_{BA}+L_M+LP_s} \right)^{\frac{1}{2}}$ .

In this analysis it has been shown that handshaking protocols do not scale well to high data rates. The main reason is the turnaround time which does not decrease in proportion to increasing data rates. On the contrary busy tone protocols with frequency duplexed feedback scale well with data rate. The next section studies the performance of two handshaking protocols and one busy tone protocol to justify the above arguments.

## 4 Performance of Current Ad-hoc MAC Protocols

### 4.1 Distributed Foundation Wireless MAC (DFWMAC)

DFWMAC is the basic access protocol in the recently standardized IEEE 802.11 wireless LAN standard [6]. It uses four way handshaking to handle the hidden nodes (RTS-CTS) and an unreliable wireless channel (ACK). When a node has data to transmit, it tries to acquire the channel by sending an RTS packet. If the RTS transmission is received without any errors, the destination node responds with a CTS packet indicating that it is ready to receive the data. The node then completes the packet transmission. If the RTS transmission results in a collision, no CTS is received. When RTS fails or the channel is sensed busy, the node backs off a random amount of time. An analytical expression was derived for the maximum throughput achieved by this protocol when the network is fully loaded [5]. It is,

$$\eta = \frac{P_s E(P)}{E(I) + P_s T_s + (1 - P_s) T_c}$$

where  $P_s$  is the probability of a successful transmission,  $E(P)$  is the mean packet length,  $E(I)$  is the mean idle period duration and  $T_c$  &  $T_s$  the durations of a collision and successful transmission respectively. We use this throughput equation

derived and verified in [5] with the parameters for the packet sizes and inter-frame spaces as specified in the IEEE 802.11 standard and vary the data rate from the specified 1Mbps to 100Mbps. The interframe spaces in this protocol have turnaround times included in them.

#### 4.2 Elimination Yield–Non Preemptive Multiple Access (EY-NPMA)

EY-NPMA is the channel access protocol used in the HIPERLAN system developed in Europe [7]. The channel access has three phases: prioritization phase (the highest priority is decided), contention phase (nodes of the same priority contend and one station wins) and transmission phase (successful station completes the data transmission) [7]. To understand the performance of this protocol as a function of data rate it is necessary to identify parameters in the protocol that scale with data-rate and those that are hardware limitations. Each component of the protocol is examined and their dependence on data rate is listed below.

- **Low Bit Rate header:** At the beginning of every data packet transmission the standard specifies a portion of the header that should be transmitted at a lower data rate so that most of the circuits need not be turned ON unless necessary. It is possible that if the MAC is upgraded to a higher data rate, the LBR header also might be proportionally increased. In this analysis however, we assume that this is not changed. ( $T_{LBR} = 34$  LBR bits =  $24\mu\text{s}$ ).
- **Prioritization Slot** ( $i_{PS}$ ) and **Priority Assertion Slot** ( $i_{PA}$ ) have a turnaround time embedded in them. A node has to sense the channel if a higher priority node is transmitting (receive mode) and if the channel is idle, the node has to turnaround and assert priority. Hence these slot sizes are hardware dependent. ( $i_{PA}=i_{PS}=168$  HBR periods =  $7.14\mu\text{s}$ ).
- **Elimination phase slot** ( $i_{ES}$ ) and **Survival Verification interval** ( $i_{ESV}$ ) also have a turnaround time embedded in them. In the elimination phase a sender transmits for a random duration of time and then has to turn around and listen for a survival verification period. Hence these slots have to be longer than one turnaround time and a sensing interval, both of which are dependent on the hardware. ( $i_{ESV} = i_{ES} = 212$  HBR periods =  $9\mu\text{s}$ ).
- The **Yield phase slot** is similar to prioritization slot ( $i_{YS}$ ) and is of the same size. ( $i_{YS} = i_{PS}$ ).
- Data is transmitted at a higher data rate and the transmission time correspondingly decreases.
- The acknowledgment transmission time decreases with higher data rate. At the end of the data transmission the receiving node has to turn around and send the acknowledgment. ( $T_{AK} = 512$  HBR periods +  $21\mu\text{s}$ ).

We can use the throughput equation derived in [4] with the parameters specified in the standard. The throughput as a function of the data rate is given by,

$$\eta = \frac{E(P)P(T=1)}{\frac{E(P)^{496}}{416} + R_D(i_{PA} + i_{ESV} + i_{AK} + i_H) + T_{LBR}R_L + R_D(i_{ES}n_E + i_{YS}n_Y)}$$



where  $E(P)$  is the mean packet size,  $P(T=1)$  is the probability that a single transmission occurs during data phase, the ratio  $\frac{496}{416}$  is due the error correcting codes specified in the standard,  $i_H$  is the synchronization interval in the packet header and  $n_E$  and  $n_Y$  are the mean lengths of the elimination and yield phases.

### 4.3 Receiver Initiated - Busy Tone Multiple Access

Receiver Initiated - Busy Tone Multiple Access (RI-BTMA), though initially proposed as a modification to BTMA to improve efficiency, was probably the first protocol that took advantage of the fact that the DN is the only node that can tell whether the current transmission is a collision or not [3]. When a node has data to transmit, it samples the busy tone channel. If the busy tone channel is idle, it initiates a data transmission. If this transmission is received at the destination without any errors, the destination station asserts on the busy tone channel. If at the end of the slot, the busy tone channel is not asserted, the transmission is aborted and rescheduled after a random period. An equation was derived for the throughput for this protocol in [3]. It is given by,

$$\eta = \frac{P_s(g+1)}{P_s+g}$$

where,  $P_s$  is the probability of exactly one packet arrival in a slot and  $\frac{1}{g}$  is the mean packet size in slots. The slot is chosen such that it is long enough to detect a collision and transmit the feedback ( $= (L_{BA} + L_{BD}(1 + \frac{1}{\alpha}) + L_M)$  bits).

## 5 Performance Results

To study the performance of these three protocols as function of data rate we need a model of how the turn-around time changes with data rate. Assuming that the turnaround time will not vary with data rate is very simplistic and probably not very realistic. At the other extreme, the turnaround time does not decrease in proportion to the increase in data rate. In this analysis, we assume a power law variation with rate.  $\tau = \frac{\tau_1}{R^{(1-\gamma)}}$  where  $\tau_1$  is the turnaround time at 1 Mbps, and  $R$  is the rate of data transmission.  $\gamma$  indicates how well  $\tau$  scales with data rate and is in the range  $[0,1]$ .  $\gamma = 0$  corresponds to the case where turnaround times scale down in proportion to increase in data rate and  $\gamma = 1$  corresponds to the case when turnaround times are independent of rate.

The efficiency of DFWMAC and EY-NPMA protocols as a function of data rate and packet size with  $\gamma = 1$  is shown in Fig. 3 and Fig. 4. The data rate is varied from 1Mbps to 100Mbps. Three different packet sizes were considered. 8184 bits is the maximum packet size specified in the 802.11 standard and 19080 bits is the maximum message size in the HIPERLAN standard. A third packet size of 1000 bits is considered because it more accurately represents the mean packet size in a typical LAN. It can be seen that the performance of both the ad-hoc wireless MAC protocols decreases monotonically as the data rate is increased. With 1000 bit packets the performance is much worse. At small packet

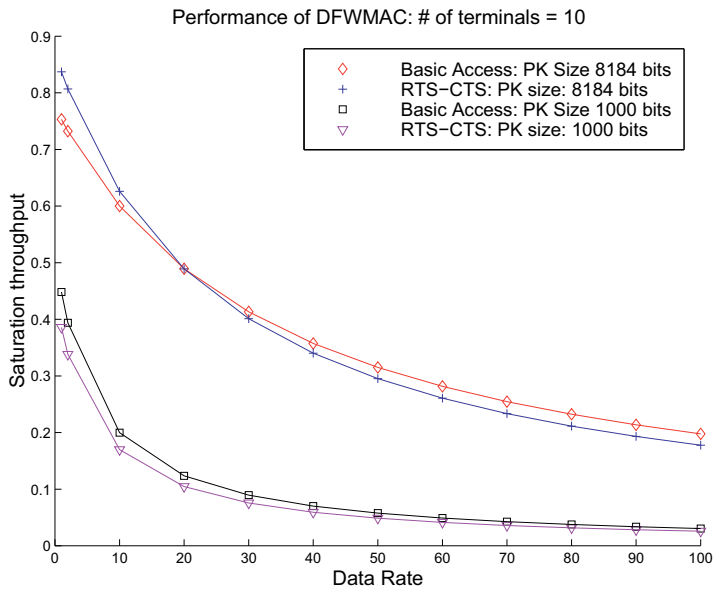


Fig. 3. Performance of DFWMAC as the data rate is increased.

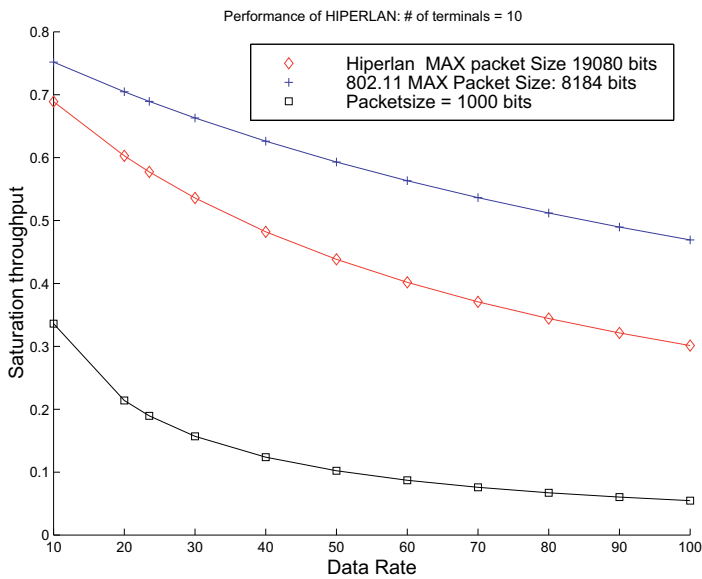


Fig. 4. Performance of EY-NPMA as the data rate is increased.

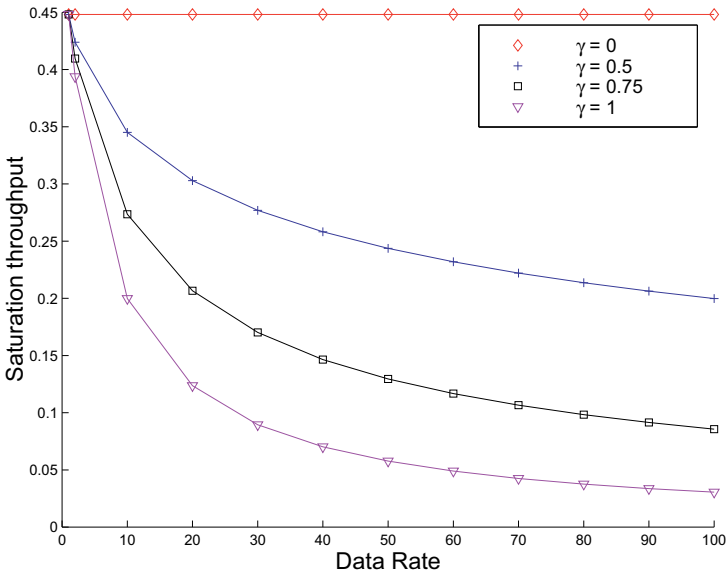


Fig. 5. Performance of DFWMAC for four different values of  $\gamma$  (0.0,0.5,0.75,1.0).

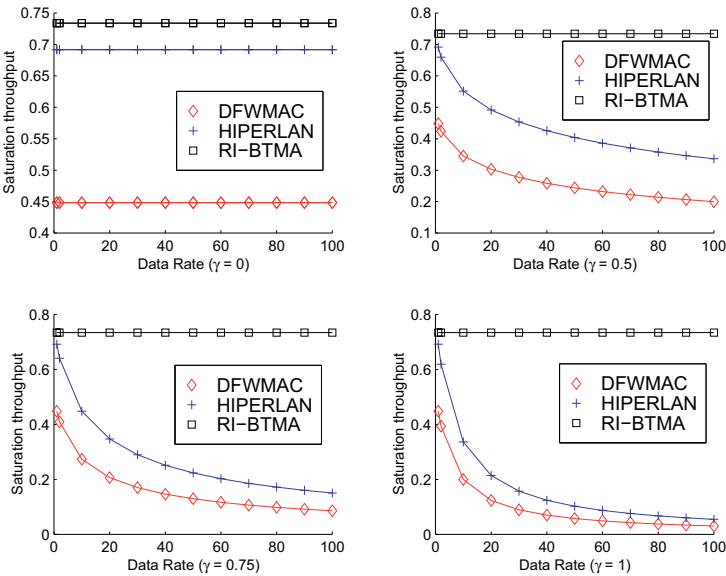


Fig. 6. Performance of DFWMAC, EY-NPMA and RI-BTMA for four different values of  $\gamma$  (0.0,0.5,0.75,1.0).

sizes the efficiency is not better than the simple Slotted-ALOHA even at the data rates at which the standards are proposed. Fig. 5 shows the performance of DFWMAC for 1000 bit messages for different values of  $\gamma$ . This indicates that the degradation in performance is significant even if the turnaround times decreased by an order of magnitude for two orders of magnitude increase in data rate ( $\gamma = 0.5$ ). Fig 6 compares the three protocols for a 1000 bit packet and different values of  $\gamma$ . This shows that RI-BTMA protocol's performance is not affected by variations in turn-around times and hence is rate scalable.

## 6 Conclusions

The method used to duplex the feedback information will have a significant impact on the performance of wireless ad-hoc protocols at high data rates. A mathematical analysis was presented to show that when turnaround switching times are involved, busy tone protocols scale well with data rate. The performance of wireless random access protocols proposed in current wireless LAN standards has been studied. We have identified parameters in these MAC protocols that have turnaround periods and hence are critical in the design of high speed MAC protocols. It was shown that as we move to higher data rates these CSMA/CA protocols perform very poorly. This performance loss can be largely attributed to the turnaround times involved in all time-duplex protocols. The results indicate that unless the turnaround times scale in proportion to data rate the performance of CSMA/CA protocols will be worse than S-ALOHA.

## References

1. P.Karn: MACA - A New Channel Access Method for Packet Radio, ARRL/CRRL Amateur Radio 9th Computer networking Conference, Sept 22,1990.
2. F.A.Tobagi and L. Kleinrock: Packet Switching in Radio Channels: Part II - The Hidden Terminal Problem in Carrier Sense Multiple Access and the Busy Tone Solution, IEEE Trans. Commun., COM-23, 1975 pp 1417-1433.
3. Cheng-Shong Wu and Victor O.K. Li: Receiver-Initiated Busy Tone Multiple Access in Packet Radio Networks, Proc Sigcomm '88 pp 336-342.
4. G. Anastasi, L. Lenzi, and E. Mingozzi: Stability and Performance Analysis of HIPERLAN Proc. IEEE INFOCOM'98.
5. Giuseppe Bianchi: IEEE 802.11 Saturation Throughput Analysis, IEEE Commun. Letters, Vol. 2 N0. 12, December 1998.
6. Brian P. Crow, Indra Widjaja, Jeong Geun Kim, and Prescott T. Sakai: IEEE 802.11: Wireless Local Area Networks, IEEE commun. mag. Sept 1997.
7. ETSI: HIPERLAN Functional Specification, ETSI draft standard, July 1995.
8. Raphael Rom and Moshe Sidi: Multiple Access Protocols Performance and Analysis, Springer-Verlag 1990.
9. Hyoung-Kee Choi and John O. Limb: Discussions on the traces collected on Gatech LAN network

# Experimental Investigation of the Effects of Delay and Error Rate on Throughput and Performance of ATM Satellite Links Serving LAN Islands

Sufian Yousef and Caroline Strange

Anglia Polytechnic University, Victoria Road South, Chelmsford, Essex CM1 1LL, UK

**Abstract.** Satellite networks, offering broad geographical coverage and deployment, appear to be an attractive option, provided a number of difficulties deriving from the nature of satellite systems can be overcome. A group of experiments has been performed using Marconi Research Centre ATM Test-bed applying the NetPIPE benchmarking tool to investigate these difficulties such as the influence of the round-trip-delay and the error rate on the throughput, delay variations and performance of LAN islands characterised by their backward low bit rate (2.048 Mbps) over ATM satellite networks. Consequently, Some characteristics are defined and some conclusions regarding the future performance of TCP/IP over satellite ATM networks, are drawn.

## 1 Introduction

ATM is rapidly emerging as the backbone of the future information networks. The ATM protocol is based on international standards and is designed to support multimedia information services. A combination of flexible bandwidth allocation, statistical multiplexing, priority queuing, and multicasting make ATM potentially more bandwidth-efficient than conventional switching technologies, despite the additional overhead of 9.4 percent for the 5-byte header per 53-byte ATM cell. Flexible bandwidth allocation is a key reason. The ability to allocate any and all of the available bandwidth on demand can outweigh the reduction in peak throughput. The peak throughput is sacrificed in favour of flexibility, with the goal of greater overall bandwidth efficiency and the ability to accommodate multimedia information services on demand. Additionally, the ability to multicast information avoids unnecessary transmission of duplicate information, such as copies of e-mail messages or video transmissions, which may be especially relevant over bandwidth-constrained satellite links.

In near future, ATM is expected to coexist with conventional networks based on IP. Logically, IP over ATM (IP/ATM) is one way to integrate ATM with legacy networks. The integration of ATM LANs into WANs via satellite links provides obvious advantages in terms of the ability to extend communications to geographically distant or remote locations, but it also poses challenges due to the longer delay, delay jitter, and potentially higher error rates. Efficient error control tailored to the ATM satellite protocol stack, is required. Interface and rate conversion

is required since such LAN-satellite links do not support rates as high as those of terrestrial fibre optic links.

Satellite systems have several inherent constraints. The resources of the satellite communication network, especially the satellite and the Earth station, are expensive; they must be used efficiently. The large delays in geo-stationary Earth orbit (GEO) systems and delay variations in low Earth orbit (LEO) systems affect both real-time and non-real-time applications. In an acknowledgement and time-out-based congestion control mechanism such as TCP, performance is inherently related to the delay-bandwidth product of the connection. Additionally, TCP round Trip Time (RTT) measurements are sensitive to delay variations that may cause false time-outs and retransmissions.

Therefore, the congestion control issues for broadband satellite networks are to some extent different from those of low-delay terrestrial networks. Both interoperability issues as well as performance issues need to be addressed before transport layer protocols like TCP can satisfactorily work over long-latency satellite ATM networks.

Broadband networking via LAN-SAT would place significant demands on already limited spectral resources at 14/12 GHz. More bandwidth is available at extremely high frequency (EHF) Ka-band. However, losses due to rain and atmospheric attenuation can be considerable at EHF, where link availability and recovery from outage may place limitations on the services that can be supported.

Broadcast or multicast of broadband video transmission to a user group is much more efficient than sending each user in the group a separate copy. A multicast transmission on the forward link can be coupled with a low-rate return link from individual users, which may be employed for error recovery of data transmission. IP multicasting over ATM networks is more practical at present. In the future, multicasting may be supported by IP version 6 over ATM.

TDMA was the chosen access scheme for satellite links. Stations transmit traffic bursts that are synchronised so that they occupy non-overlapping time slots which are organised within periodic frames. Each station can extend its traffic from the down-link.

The satellite has a link capacity of 25 Mbps (36 Mhz bandwidth) per transponder at present on Eutelsat II. The satellite link capacity has to be shared by a number of earth stations making a multiple islands interconnection. It is required for the satellite communication to provide the required QOS with an efficient utilisation of the satellite resources. The delay variation is done through the buffering in the ground segment due to the traffic load on the buffer. Most of the delay variation is at Terrestrial Interface Module for ATM (TIM-ATM) buffers. Cell loss occurs when buffer overflows. Ideally controlling the number of applications, traffic load and adequate bandwidth allocation for each application can control cell loss and the delay variation.

The frame duration is designed to be so small to produce reasonable constant delays due to frame length transmission. The RTT specified by the telephone services is at least 400 ms, taking into account the terrestrial (130 ms) and propagation delays (270 ms). The chosen frame size as 20 ms is meant to give acceptable performance. As the satellite header is normally long, the granularity of the capacity assignment to the required bit rate is increased through decreasing the frame length [1].

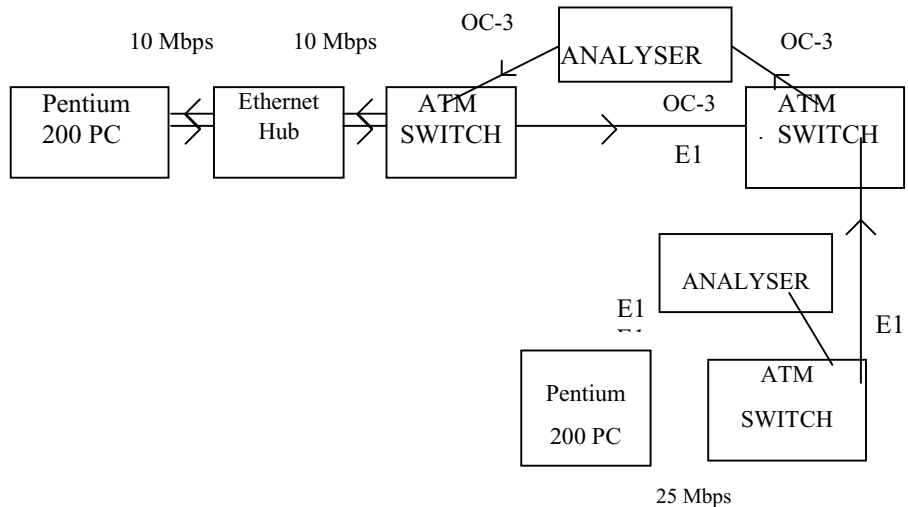
This scenario is similar to VSAT satellite system typical of corporate networking and rural communications. A large number of small dishes of 3.4 metres diameter stations carrying traffic of bursty nature, need to be served by the satellite. The multiple access is the major problem due to that we deal with various classes of traffic where compatibility of ATM format is needed to be provided (CBR, ABR, UBR, etc).

However, a concept for combining the benefits of ATM with those of SATCOM at Ka-band and error control concepts tailored to ATM SATCOM are proposed by [2]. The point which was not clear is what is sensitive to error rate in TCP/IP transmission and why. Also, the effect of delay on throughput and the effect of various data block sizes when combined to the effect of error rate on delay variations were missing.

This rest of the paper is organised as follows. In section 2, the experiment for the RTT influence on the performance of the low bit-rate islands at 2.048 Mbps using NetPIPE bench marking tool is presented. The effect of delay on throughput is included in section 3. In section 4, the effect of bit error rate on throughput and delay Variations is investigated. The paper is concluded in section 5.

## 2 Imposing Delay and Error Rate Experiment Using NetPIPE

The aim of this experiment was to impose different RTT values on the low bit-rate TCP and UDP/IP generated traffic on the simulated network to measure their influence on the throughput, block transfer time and its variance at different block sizes and window socket sizes. The results were represented in specific network performance graphs.



**Fig. 1** Set-up for the Simulation of Ground and Satellite Segments

This experiment was performed using NetPIPE benchmark tool [3] for TCP/IP and Netperf [4] for UDP/IP traffic. The transmitting command was originated from a

Pentium 200 MHz PC connected to a LAN emulation Network Interface Card (NIC). The receiving command was originated from another Pentium 200 MHz PC connected to an ATM NIC. The E1 bit-rate was chosen to represent traffic originating from a distributed LAN island communicating with an ATM satellite network similar to Sat-Com network.

The testing set-up shown in Fig. 1 simulates a ground and satellite network segments. The generated TCP, UDP/IP traffic originated in a 25 Mbps bit-rate. Then, it is directed towards the ATM switch [5] in 2 Mbps bit-rate. The satellite link is simulated by generating signal towards the Analyser where a chosen delay is imposed on an OC3 (155 Mbps) bit-rate and then towards the receiving ATM switch (Earth Station). This traffic was then forwarded towards an Ethernet hub on an E1 (2.048 Mbps) bit-rate and then to a receiving PC.

The receiving side of traffic was forwarded from a PC towards the ATM switch via the Ethernet hub and then towards the ATM switch on OC3 bit-rate. From this ATM switch it headed towards the other ATM switch on an E1 (2.048 Mbps) bit-rate through the analyser which was used to impose the testing required delay on the E1 bit-rate. Traffic is transferred back from the ATM switch to the PC on a 25 Mbps bit-rate.

The testing procedure requires the use of NetPIPE and Netperf benchmarks as well as the analyser.

The experiment is repeated several times such that the total time for the experiment is far greater than timer resolution. NetPIPE uses a ping-pong transfer for each block size. This forces the network to transmit just the data block without streaming other data blocks in with the message. The result is the transfer time of a single block, thus providing the information necessary to know which block size is best, or what is the throughput given a block of size k. a sample of the result files is shown in Table. 1.

**Table 1.** A sample of TCP over ATM Satellite Network Performance at Default Socket Size.

Number of bytes	Time to transfer block in s	Throughput in Mbps	Variance
1	0.2045	0.000037	0.000001
128	0.2052	0.0.0048	0.000003
512	0.2054	0.019	0.000004
1024	0.211	0.038	0.000004
2048	0.214	0.073	0.000006
4096	0.253	0.123	0.00218
6144	0.320	0.147	0.00255
8192	0.653	0.096	0.004443
12288	0.719	0.131	0.00438
16384	0.951	0.132	0.016791
24576	1.257	0.149190	0.012120



Due to the high delay imposed (400 ms RTT) NetPIPE was unable to measure block sizes > 24 Kbytes. Measurements were taken on the end-to-end configuration. Traffic of 2.048 Mbps was chosen from Ethernet-to-Ethernet which represented LAN islands interconnected via satellite systems using ATM.

The imposed RTT delay is 400 ms on the satellite connection. Other delay values such as 100, 50, 20, 10, 1ms are imposed to simulate respectively Medium Earth Orbit (MEO), Low Earth Orbit (LEO), terrestrial wireless links, mobile links and optical fibre links.

Delay Variation is an important factor in satellite communication. It depends on the data block sizes and the retransmissions of the Ethernet LAN. This delay variation is found empirically to be equal (TDMA Frame Time / 2). But in this report we managed to measure the delay variation accurately through the measured variance of the block transfer time by using NetPIPE.

The Request and Response Time = Time to generate Request + Time to process Request + RTT.

The Buffer Requirement (BB) in the satellite link applying ATM cells is computed as follows:  $BB = (\text{Trans speed} - \text{Link speed}) * (\text{Burst size} / (53 * \text{Trans speed}))$ .

Throughput is restricted by the wait for the acknowledgements. The window size of the protocol can be used to adjust the amount of data to be sent before waiting for the acknowledgement. The following empirical formula estimates the throughput of burst traffic based on connection-oriented protocols with acknowledgements:

$\text{Throughput} = \text{Window size} / (\text{Transmission rate} * \text{RTT})$ .

For example: Suppose that the transmission rate is 2.048 Mbps and RTT is 0.5 second. The window size =  $\text{Transmission rate} * \text{RTT} = 2.048 \text{ Mbps} * 0.5 = 1.024 \text{ Mbps} = 128 \text{ k bytes}$ . This means that the throughput is 100% if we ignore the overheads. Throughput is directly proportional to the window size. If throughput is measured as 0.8 Mbps, the Acknowledgement window size is 400 k bits which is approximately 52 k bytes. In our measurements in Marconi Research Centre (MRC) the default socket size on NetPIPE was 8192 bytes. This is why the maximum measured throughput was 8.192/128. To convert the measured throughput to any other socket value we need to multiply it with  $X/128$  where  $X$  is the socket size in k bytes. All measurements in this experiment have taken into account a default error rate of  $1 \times 10^{-12}$ .

### 3 The Effect of Delay on Throughput

In Fig. 1, the satellite network signature graph is shown. This graph shows the network latency when 1 byte is transferred and the effect of the buffer on the throughput when the transfer time increases. It is apparent that at small transfer times (0.33 s), both buffer lengths have the same throughput. Beyond the 0.33 s transfer time point in the graph, throughput increases for the larger buffer size. This shows how larger buffer sizes are important in improving throughput at larger message sizes and consequently larger transfer times.

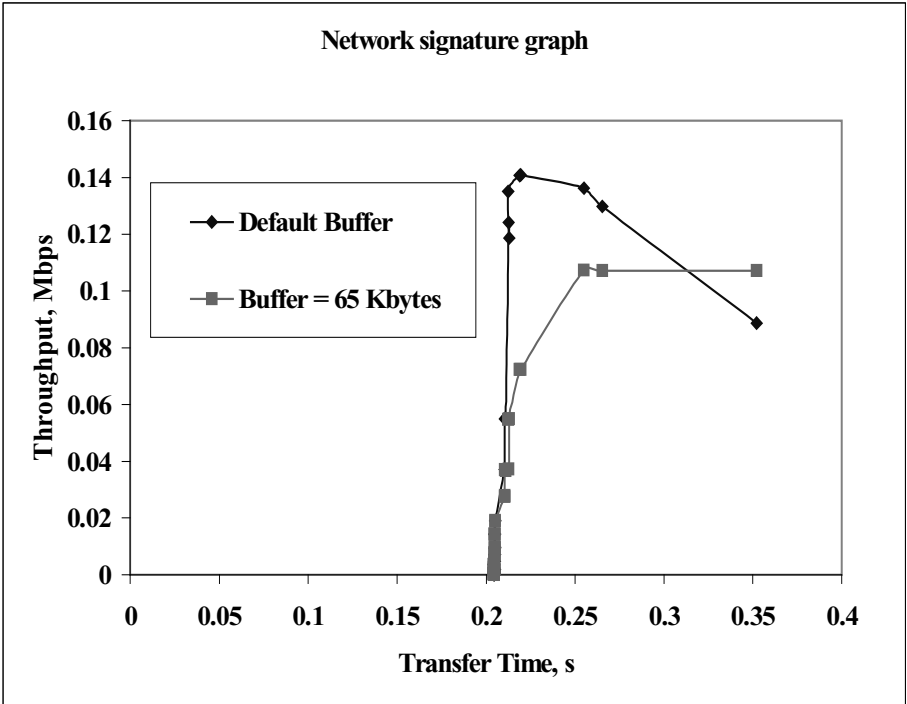
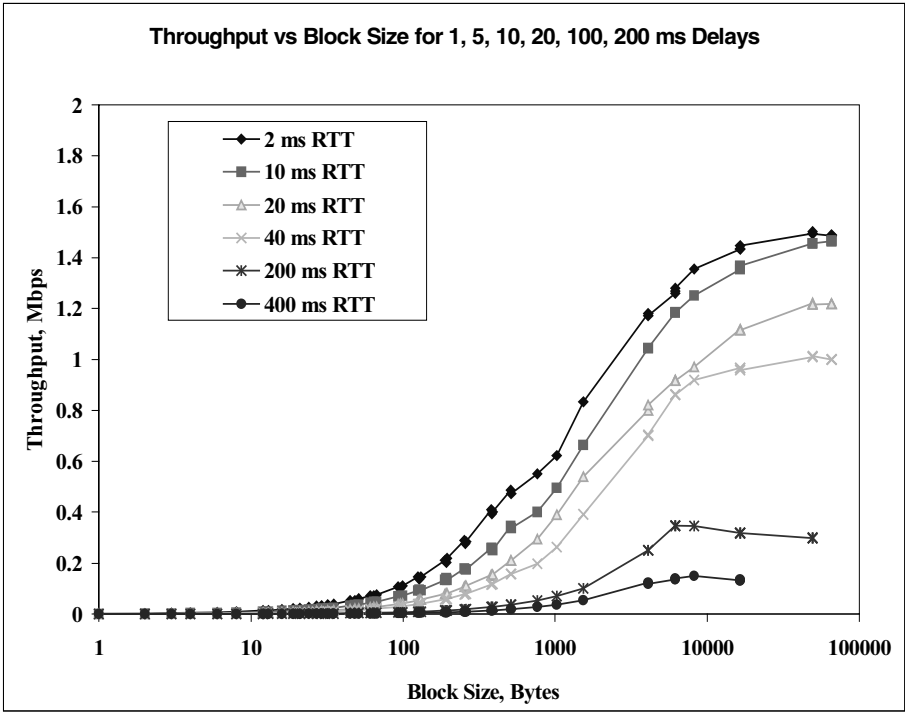


Fig. 1. Satellite Network Signature Graph

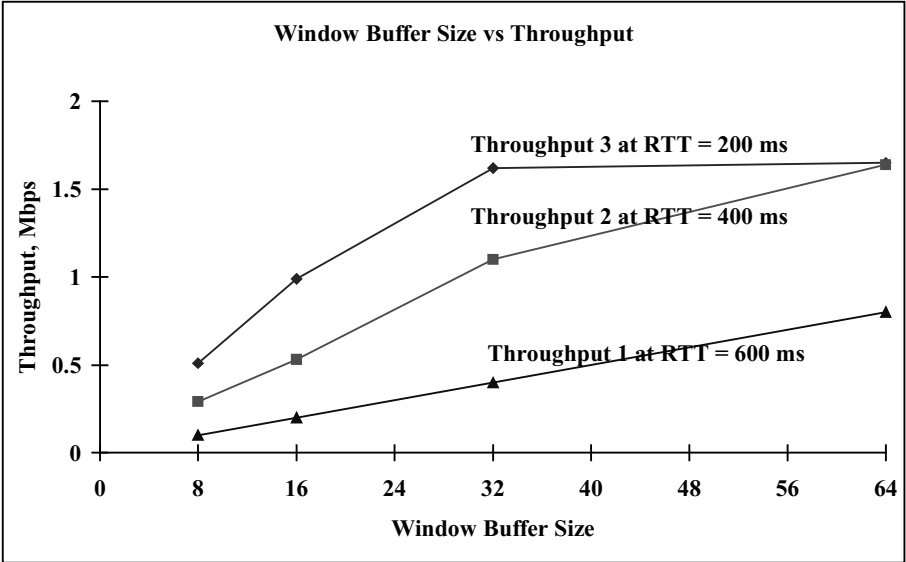
In Fig. 2 the effect of the round trip time (RTT) is shown. Normally increasing the block size increases the throughput. But the slope of throughput versus block size is different at different RTT values. The maximum throughput can be achieved when RTT is around 1 to 2 ms (short haul radio links). The 20 ms delay is typical of ATM wireless links where throughput is reasonably good. At Low Orbit military satellite links where delay is 40 to 50 ms, throughput declines abruptly but the maximum block size (65 k bytes) is still appears in the output if required to be processed which makes the Low Orbit satellites unique in this characteristic if compared with other satellite types. At medium orbit satellite links where RTT is around 200 ms throughput drops down and hence other factors such as the buffer size and the socket size has to be taken into account to compensate for this throughput decline. However, the maximum block size to be transferred is upper bounded at around 49 k bytes if such data block size has to be processed. At the traditional GEO high orbit satellite (34000 km) where the RTT value is  $\geq 400$  ms, the throughput is very small comparatively and the maximum data block size in bytes is limited to 16 k bytes if it is to be processed.

The effect of the buffer size on throughput is shown in fig. 3. Throughput increases with the increase in the socket buffer size at different slopes depending on the satellite link RTT. At low orbit satellites of  $RTT \leq 100$  ms the throughput increases in a faster slope until the socket size is 32 k bytes. Beyond this socket size throughput becomes nearly constant until it reaches the maximum block size (64 k bytes). At the medium

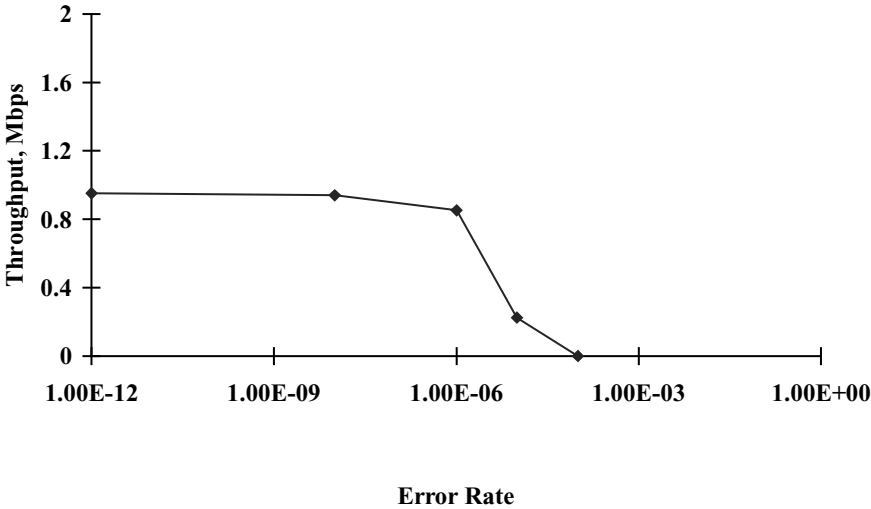
orbit satellites of  $RTT \leq 400$  ms throughput increases quite linearly on the socket size range and it joins the low orbit throughput at the maximum socket size (64 k bytes). At the RTT value of 600 ms, throughput increases linearly but in smaller slope at the whole socket size range. This figure helps the design engineer to allocate a specific socket size to target a specific throughput. This Figure shows that the allocation of the default socket buffer size is impractical when throughput is demanded. Implementing TCP/IP over ABR is shown by the NetPIPE to have a linear relationship at very long RTTs such as 600 ms which is typical of Geo-stationary satellites. This means that the window buffer size on very long RTTs depends on the throughput-delay product only which yields buffer size in bits out of this product:  $Mbps \times \text{Time in s} = M$  buffer size in bits =  $M/8$  buffer size in bytes. The formula at  $RTT = 600$  ms according to Fig. 3 is:  $\text{Buffer Size} = \text{Throughput} \times RTT$ . In other words, the equation representing the straight line for this linear relationship is  $y = 0.0125 x$  where  $y$  is the throughput in Mbps and  $x$  is the socket buffer size in k bytes. In LEO satellites this linear relationship is not valid where much smaller socket buffer size is required to achieve the same goal. Although this is a favourable characteristics in Leo satellite, it removes the influence of the socket buffer size in improving the delay variations.



**Fig. 2.** Block Size in bytes versus Throughput in Mbps at Different RTT Values and at a Default Socket Size.



**Fig. 3.** Window Size in K bytes versus Throughput in Mbps at Different Delay (one direction) Values

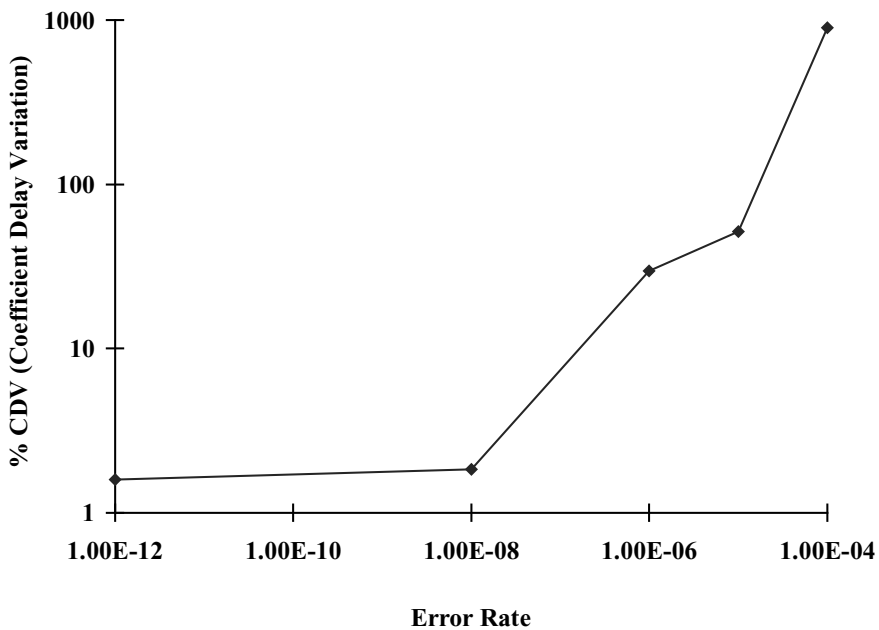


**Fig. 4.** Error Rate versus Throughput at RTT = 400 ms.

#### 4 The Effect of Bit Error Rate on Throughput and Delay Variations

Normally throughput is affected by the delay due to the constant bandwidth-delay product in the absence of bit error rate. However, the error rate experimental influence on throughput of high-orbit (GEO) satellite links is measured and shown in Fig. 4. This Figure shows a gradual decline of throughput due to the error rate on the error rate ranges from 0 to  $1 \times 10^{-6}$ . This influence is extremely higher when the error rate increases from  $1 \times 10^{-6}$  to  $1 \times 10^{-4}$ . Actually, throughput disappears when the error rate is  $1 \times 10^{-4}$ . Therefore, it is important to protect satellite links from error rates  $> 1 \times 10^{-6}$  by applying diversity or coding techniques.

In Fig. 5 the influence of error rate on the percentage CDV at default block size and RTT = 400 ms is shown. It shows the exact way the % CDV increases with the error rate. While the error rate range from  $1 \times 10^{-12}$  to  $1 \times 10^{-8}$  has a small effect on the % CDV, the range  $1 \times 10^{-8}$  to  $1 \times 10^{-5}$  has a high slope increase on the % CDV and a drastic slope increase on the error range of  $1 \times 10^{-5}$  to  $1 \times 10^{-4}$ . The lesson learned here is that for the sake of controlling the % CDV, the error rate has to be limited to  $1 \times 10^{-8}$  as a maximum possible error rate.



**Fig. 5.** Error Rate versus the Percentage Coefficient Delay Variation at RTT = 400 ms and a Default Block Size (8192 bytes).

## 5 Conclusions

The window buffer size has to be  $> 65$  k bytes to get reasonably acceptable throughput at large delay satellite networks carrying TCP/IP over ATM.

LEO military satellites RTT (40-50) ms is the best to allow the transmission of block sizes as high as 65 k bytes. Geo-stationary satellites limit the maximum block sizes to nearly 16 k bytes only. MEO satellites limit the maximum block sizes to nearly 49 k bytes. Throughput is very low at GEO-stationary satellites and hence other means have to be adopted to improve the throughput such as using selective acknowledgements (SACK Ack.), higher window buffer sizes and lower rates than 2 Mbps in the backward path of the LAN satellite systems.

Implementing TCP/IP by the NetPIPE has shown a linear relationship at very long RTTs such as 600 ms which is typical of Geo-stationary satellites. This is supported by the network graphs where it is apparent that at small transfer times, different buffer lengths have the same throughput. Beyond the intersection transfer time point in the network signature graph, throughput increases for the large buffer sizes as compared to the default buffer size. This shows how larger buffer sizes are important in improving throughput at larger message sizes and consequently larger transfer times. This result simplifies the use of the buffer to improve throughput due to the achieved empirical formula. As a result, switches connected to satellite loops only need to have buffers proportional to the bandwidth-delay products only.

Delay variations depend on the data block sizes, the bit error rate and the RTT. Bit error rate can degrade the throughput and the effect is different at different block sizes and window buffer sizes.

The accepted threshold of the bit error rate on throughput, delay variations and block transfer time is  $1 \times 10^{-6}$ . The superior low percentage coefficient of delay variations (% CDV) can be achieved on error rates of  $\leq 1 \times 10^{-8}$ .

Diversity as a powerful error control technique, may be employed to improve satellite link availability and reduce power requirements at different rain margins.

In order to minimise the transmission delay effect and support real time applications efficiently, the adjustment of existing protocols or development of new ones is required. The value of a time-out parameter or window size of protocol which requiring acknowledgements of cell arrival might be increased to accommodate the long propagation delay.

## References

- 1 M. H. Hadjitheodosiu et al., "End-to-end Performance Analysis for Broadband Island Interconnection via Satellite," RACE 11/R2074 (CATALYST) Project, European Conference of Satellite Communication (ESC-3) Project, European Conference of Satellite Communications (ECSC-3), Nov. 1993, pp. 170-174.
- 2 J. Farserotu and A. Tu, "TCP/IP over Low Rate ATM-SATCOM Links," IEEE MILCOM '96, Mclean, VA.
- 3 NetPIPE, <http://www.ameslab.gov/netpipe>.
- 4 Netperf, <http://www.cup.hp.com>.
- 5 FORE Systems, <http://www.fore.com>.

# Dynamic Algorithms with Worst-Case Performance for Packet Classification

Pankaj Gupta<sup>1</sup> and Nick McKeown<sup>1</sup>

Computer Systems Laboratory, Stanford University  
Stanford, CA 94305-9030  
{`pankaj`, `nickm`}@stanford.edu

**Abstract.** Packet classification involves — given a set of rules — finding the highest priority rule matching an incoming packet. When designing packet classification algorithms, three metrics need to be considered: query time, update time and storage requirements. The algorithms proposed to-date have been heuristics that exploit structure inherent in the classification rules, and/or trade off one or more metrics for others. In this paper, we describe two new simple dynamic classification algorithms, *Heap-on-Trie* or *HoT* and *Binarysearchtree-on-Trie* or *BoT* for general classifiers. The performance of these algorithms is considered in the worst-case, i.e., without assumptions about structure in the classification rules. They are also designed to perform well (though not necessarily the “best”) in each of the metrics simultaneously.

## 1 Introduction

Internet routers perform packet classification to identify the flow to which arriving packets belong, and hence the action or service that the packet should receive. More formally, packet classification can be defined as:

*Packet Classification:* Given a classifier with  $N$  rules,  $\{R_k\}_{k=1}^N$ , where rule  $R_k$  consists of three entities: (1) A  $d$ -tuple of ranges  $([l_1^k : r_1^k], [l_2^k : r_2^k], \dots, [l_d^k : r_d^k])$ , (2) A number indicating the *priority* of the rule in the classifier, referred to as  $pri(R_i)$ , and (3) An *action*, referred to as  $action(R_i)$ : for an incoming packet  $P$  with the relevant fields considered as a  $d$ -tuple of points  $(P_1, P_2, \dots, P_d)$ , the  **$d$ -dimensional packet classification problem** is to find a specific value of  $j$ , say  $j^*$ , such that  $pri(R_{j^*}) > pri(R_j) \forall j \neq j^*$  and  $l_i^j \leq P_i \leq r_i^j, \forall i : 1 \leq i \leq d$  in order to identify  $action(R_{j^*})$  to be applied to the packet  $P$ .

Packet classification functions have started to appear in routers to provision services [1] such as access control in firewalls, load balancing across web servers, policy-based routing, virtual private networks, network address translation, quality of service differentiation, and traffic accounting and billing.

We evaluate a packet classification algorithm for a classifier with  $N$  rules on the basis of the following metrics: (1) *Query time* – the amount of time taken to classify each arriving packet, also called the lookup or search time; (2) *Storage requirement* – memory required by the data structure used by the algorithm to hold the classification rules; (3) *Update time* – the amount of time needed to

incrementally update the data structure on insertion or deletion of classification rules. An algorithm that supports incremental updates is said to be *dynamic*. In contrast, the whole data structure has to be recomputed in a *static* algorithm, whenever a rule is added or deleted.

Fast update time is important in many applications – for example, rules need to be inserted or deleted online as flows become active or inactive in a router providing flow-based quality of service. If there is a large discrepancy between the update and query times of an algorithm, incoming packets may need to be buffered before lookup, if an update operation is in progress. Therefore, in order to avoid buffering, delay variation and head-of-line blocking problems, we seek algorithms with update time comparable to the query time.

In this paper, we present two novel dynamic algorithms and their worst-case query time, update time and storage complexities for general classifiers. These algorithms are primarily of interest in that they simultaneously have attractive worst-case complexities for each metric for any set of classification rules.

## 2 Related Work

The simplest algorithm is a linear search which sequentially compares the packet with each rule in turn, starting with the highest priority rule. The query time and storage complexities are both  $O(N)$ . Updates can be made incrementally by a simple binary search in the sorted list of rules in  $O(\log N)$  time. Thus, linear search is an example of an algorithm that performs well in two metrics (storage and update time), but poorly in the query time metric. This makes the algorithm impractical for all but the smallest set of rules.

Frequently, there is a trade-off between the query and update times. Algorithms typically achieve fast query times by pre-computation to carefully organize the data structure. This, in turn, usually renders updates inefficient. Examples of solutions that ignore update times in order to have fast query times in reasonable amount of space include a ternary CAM based solution and the bitmap intersection approach of Lakshman and Stiliadis [2]. Update time complexity is  $O(N)$  in each case.

Other solutions such as Grid-of-Tries proposed by Srinivasan et al [3], and the fractional cascading solution by Lakshman and Stiliadis [2] are static, and so are not relevant here as we are only considering dynamic algorithms. Heuristic solutions which attempt to take advantage of the structure of real-life classifiers are proposed in [3][4][5][6]. While these solutions seem to work well with real-life classifiers today, they have prohibitive  $O(n^2)$  or higher storage requirements in the worst-case. A notable exception is the tuple space search scheme proposed by Srinivasan, Suri and Varghese [7]. This scheme has  $O(N)$  worst-case storage requirement, but queries and updates can require  $O(N)$  hashed memory accesses in the worst case.

Two dynamic algorithms with sub-linear worst-case bounds have been recently proposed. The first algorithm, proposed by Buddhikot, Suri, and Waldvogel [8] uses a novel prefix-partitioning technique which helps implement fast



incremental updates. This scheme achieves  $O(\alpha W)$  search time,  $O(N)$  space and  $O(\sqrt[l]{N})$  update time where  $\alpha$  is a tunable integer parameter greater than 1. However the scheme does not readily extend to more than two dimensions or to general rules that have non-prefix field specifications. The second algorithm, proposed by Feldmann and Muthukrishnan [9], is based on a novel data structure, which they call an *FIS* tree. An FIS tree is similar to an inverted multi-ary segment tree and is essentially a static data structure. The authors also extend it to handle incremental updates. However, they state results for only a small number of updates ( $O(n^{1/l})$  in one dimension). Their scheme in one-dimension has storage complexity of  $O(n^{1+1/l})$ , update time complexity of  $O(\ln^{1/l} \log n)$  and lookup complexity of  $O(\log^2 n) + l$  memory accesses where  $l$  is a constant suitably chosen to trade-off lookup time versus storage requirement. The authors also make suggestions on how to handle larger number of updates, but have to “sacrifice” either space or lookup or update time to achieve that.

**Table 1.** Worst-case bounds obtained in this paper for dynamic  $d$ -dimensional packet classification.

Algorithm	Query	Space	Update
<i>Heap-on-Trie (HoT)</i>	$O(\log^d N)$	$O(N \log^d N)$	$O(\log^{d+1} N)$
<i>Binarysearchtree-on-Trie (BoT)</i>	$O(\log^{d+1} N)$	$O(N \log^d N)$	$O(\log^d N)$

In this paper, we focus on algorithms rather than implementation details, even though we favor readily-implemented data structures and algorithms. For a discussion of the implementation-related goals of a practical algorithm, please see references [2] and [4]. We propose two dynamic algorithms for one-dimensional classification with arbitrary classifiers and extend them to higher dimensions. We obtain the worst case bounds for both algorithms as shown in Table 1. To the best of our knowledge, these are the first published simultaneous worst-case bounds for search time, update time and space consumption for general multi-dimensional classifiers, where none of the worst-case metrics is “sacrificed” in favor of the other two.

### 3 Heap-on-Trie (HoT)

#### 3.1 One-Dimensional Classifiers

If the field specifications in a one-dimensional classifier are restricted to prefixes, a trie is a good dynamic data structure supporting inserts, deletes and searches in  $O(W)$  time, where  $W$  is the width of the field in bits, and therefore the depth of the trie (see [10] for an example of algorithms with a trie-like data structure). The space complexity of the trie is  $O(NW)$ .

If the field specifications in the rules of a classifier are not restricted to be prefixes but allowed to be arbitrary contiguous ranges, one method of storing a rule is to split a range into several maximal prefixes and to then use a trie. For example, a range  $[0, 10]$  (with  $W = 4$ ) can be split up into three maximal prefixes: (1)  $0^{***}$  denoting the range  $[0, 7]$ ; (2)  $100^*$  denoting the range  $[8, 9]$  and (3)  $1010$  denoting the single element range  $[10, 10]$ . Note that any range in  $[0, 2^W - 1]$  can be split into a maximum of  $(2 * W - 2)$  such prefixes. We call the set of constituent prefixes of a range,  $G$ , as  $C(G)$ . We say that a range  $G$  is allocated to a particular trie node  $z$  if  $C(G)$  has a prefix represented by the trie node  $z$ .

Some constituent prefixes of different ranges might be identical because ranges can overlap. Hence multiple rules may be allocated to the same trie node. In a static solution one simply pre-computes the highest priority of these rules and store it in the corresponding trie node. However, this is unacceptable in a dynamic solution where rules can be inserted and deleted because the identities of individual rules need to be maintained. The *Heap-on-Trie* data structure uses a heap at each trie node to maintain the rules allocated to that trie node. A heap is a data structure for storing keys with the following property – the key value at every non-leaf node is higher than the key values of its two children nodes.

A *Heap-on-Trie* (HoT) is therefore a two-level data structure where the set of ranges associated with a particular node is arranged in a heap, ordered by the priority of the ranges. The heap property ensures that the maximum priority of all the rules associated with a trie node is stored in the root node of the heap at that trie node and is hence available in  $O(1)$  time. Given this data structure, a classification query proceeds downwards starting from the root of the trie, returning the maximum priority rule stored at the root node of each of the heaps associated with the nodes traversed in the trie. The total query time is  $O(W)$ . A given range to be inserted or deleted is first split into  $O(W)$  prefixes. Each of these prefixes is then inserted/deleted separately to/from the corresponding heaps. The heaps are then readjusted to maintain heap order such that the highest priority range is available in its root node. Since an insert or delete takes  $O(\log N)$  time in a heap, the total update time is  $O(W \log N)$ .<sup>1</sup> The total storage requirement is  $O(NW)$  because there are  $O(W)$  constituent prefixes of a range and the total number of trie nodes that can be contributed by these constituent prefixes is also  $O(W)$ . Hence each range consumes  $O(W)$  space for a total complexity of  $O(NW)$ . Note that this is the same space complexity as a classifier with prefix-only field specifications.

Simplifications to the data structure are possible in restricted environments. For example, when it is known that overlaps among ranges are small, we could simply maintain a linked list at each node instead of a heap. Similarly, if an application is such that only inserts need to be supported, once the data structure

<sup>1</sup> For the delete operation, we need to have access to the pointers to the  $O(W)$  nodes in different heaps containing the constituent maximal prefixes of a particular range. This can be easily maintained in a separate table indexed by the rule identifier at an extra cost of  $O(NW)$  space.

has been built, insert time can be decreased to  $O(1)$  at each node by simply checking the priority in front of the list. We insert the new rule in front of the list if this is lower than the priority of the new rule, and after the first rule otherwise. This takes a total of  $O(W)$  time for a rule insertion. Likewise, if only deletes need to be supported, time for a delete operation could be brought down to  $O(W)$  by using a sorted list instead of a heap and storing the pointers to the constituent prefixes of a range in different lists corresponding to each range present in the classifier. To delete a range, one would access the corresponding prefixes by these pointers and delete them from their respective sorted lists in  $O(1)$  time each, to obtain a total delete time of  $O(W)$ . A summary of these bounds is shown in Table 2.

**Table 2.** Bounds for the different types of dynamic algorithms using the HoT data structure.

Algorithm-HoT	Query	Space	Insert-Time	Delete-Time
<i>Dynamic</i>	$O(W)$	$O(NW)$	$O(W \log N)$	$O(W \log N)$
<i>Semi-dynamic (only inserts)</i>	$O(W)$	$O(NW)$	$O(W)$	–
<i>Semi-dynamic (only deletes)</i>	$O(W)$	$O(NW)$	–	$O(W)$

### 3.2 Multi-dimensional Classifiers

The dynamic data structure for  $d$ -dimensions can be obtained by building hierarchical multi-level tries, one level for each dimension except for the last, on which a *Heap-on-Trie* is built. It is not difficult to see that the total space consumed is then  $O(NW^d)$  with query and update times being  $O(W^d)$  and  $O(W^d \log N)$  respectively. We do not use fractional cascading in these multi-level tries, as it is essentially a static technique for searching among similar lists. Mehlhorn and Naher have proposed a dynamic version of fractional cascading [11] where updates to cascaded lists can be made such that query time complexity only increases from  $O(\log N)$  to  $O(\log N \log \log N)$ . However, the constants hidden in the  $O(\cdot)$  notation are so high that a simpler  $O(\log^2 N)$  solution that does not use fractional cascading is usually better for all practical values of  $N$  [12].

## 4 Binarysearchtree-on-Trie (BoT)

The motivation in this section is to develop an algorithm that executes updates faster than the *HoT* algorithm described above. As is often the case, faster updates are obtained at the expense of increased query time.

### 4.1 One-Dimensional Classifiers

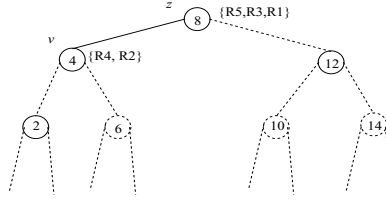
We start with defining the terminology used in this section and then describe the *BoT* data structure and algorithms.

#### Definitions

- For a node  $z$  in a trie  $T$ , we let  $parent(z)$ ,  $lchild(z)$  and  $rchild(z)$  denote its parent, left and right children nodes. Also, we let  $root(T)$  denote the root node of trie  $T$ .
- For a node  $z$  of a  $W$ -bit trie, we define  $prefix(z)$  to be the prefix that  $z$  represents in the trie. The length of  $prefix(z)$ , denoted as  $length(prefix(z))$ , is defined to be the distance of the node  $z$  from the root of the trie. We define the root's children to be at a distance of 1 from it. We can write  $prefix(z)$  by the  $W$ -bit string  $z_1z_2 \dots z_l * \dots *$ , where  $l = length(prefix(z))$ , or simply by an  $(l+1)$ -bit string  $z_1z_2 \dots z_l*$  where the trailing  $(W-l)$  ‘\*’ wildcard-bits are implicit.
- $prefix(z)$  defines a contiguous range, called  $prefixRange(z)$  of size  $2^{W-length(prefix(z))}$  on the integer number line. The starting point of this range is denoted by  $st(z)$  and equals  $z_1z_2 \dots z_l0 \dots 0$ , while the ending point of this range is denoted by  $en(z)$  and equals  $z_1z_2 \dots z_l1 \dots 1$ . Thus,  $prefixRange(z) = [st(z), en(z)]$ .
- We associate a distinguished point,  $Pt(z)$ , with every trie node,  $z$ , and refer to it as the *d-point*. The d-point is the midpoint of  $prefixRange(z)$ . Equivalently,  $Pt(z) = \lfloor (st(z) + en(z))/2 \rfloor = z_1z_2 \dots z_l10 \dots 0$ .
- $G(z)$  denotes the set of ranges allocated to trie node  $z$ . The algorithm for allocation of ranges to nodes is described below.
- $PGL(z)$  (respectively  $PGR(z)$ ) is defined to be the set of the leftmost (rightmost) endpoints of the ranges in  $G(z)$ . If  $e$  is any such endpoint in  $PGL$  or in  $PGR$ , we let  $range(e)$  denote the range for which  $e$  is one of the endpoints.

**Table 3.** An example one dimensional 4-bit classifier consisting of arbitrary ranges. The priority of  $R_i$  is assumed to be  $i$ .

Rule	Range	Maximal Prefixes
$R_5$	[3,11]	0011, 01**, 10**
$R_4$	[2,7]	001*, 01**
$R_3$	[4,11]	01**, 10**
$R_2$	[4,7]	01**
$R_1$	[1,15]	0001, 001*, 01**, 10**, 110*, 1110



**Fig. 1.** Showing the range allocations to the trie nodes for rules in Table 3. The number inside a trie node  $v$  represents  $Pt(v)$  associated with it. In this particular example, all ranges are allocated to the root node and its left child. The remaining nodes are actually not present in the trie – they are only shown here to illustrate the calculation of the distinguished points.

**Table 4.** Showing the allocated ranges to the trie nodes.

Trie Node, $w$	$Pt(w)$	$G(w)$	$PGL(w)$	$PGR(w)$
$z$	8	$\{R_5, R_3, R_1\}$	$\{1, 3, 4\}$	$\{11, 11, 15\}$
$v$	4	$\{R_4, R_2\}$	$\{2, 4\}$	$\{7, 7\}$

**The *BoT* Data Structure** Similar to *HoT*, we use a trie as the underlying data structure. The basic difference is that we now allocate a range to only one trie node rather than to  $O(W)$  nodes. We allocate a range  $H$  to a trie node  $z$ , if  $z$  satisfies the following two conditions: (1)  $H$  contains  $Pt(z)$ ; and (2) If  $z$  is not the root node,  $H$  does not contain  $Pt(parent(z))$ . For example, range  $R_5[0011, 1011]$  in Table 3 is allocated to the root node of the trie as it contains the point  $1000 = Pt(root(T))$ ; while range  $R_4[0010, 0111]$  is allocated to the left child of  $root(T)$  as it does not contain  $Pt(root(T))$  but contains  $Pt(lchild(root(T))) = 0100$ . We allocate every range in the one-dimensional classifier to exactly one trie node in this manner. The set of allocations for the example classifier in Table 3 is shown in Figure 1. This is also shown in tabular form along with the sets  $PGL$  and  $PGR$  in Table 4.

Each of the sets  $PGL(z)$  and  $PGR(z)$  is stored in a balanced binary search tree (BBST) —  $PGL(z)$  in  $BBST_{left}(z)$  and  $PGR(z)$  in  $BBST_{right}(z)$ . We can choose one of various implementations of a BBST data structure, such as red-black trees [13] and 2-3 trees [14] for  $BBST_{left}(z)$  and  $BBST_{right}(z)$ . We have three fields in a node,  $x$ , of the BBST:

1.  $val(x)$  which stores the value of one of the endpoints of the relevant range. Nodes in  $BBST_{left}$  contain the left endpoint and those in  $BBST_{right}$  contain the right endpoint of the range associated with the node.
2.  $pri(x)$  which stores  $pri(range(val(x)))$ .
3.  $augp(x)$  which augments the BBST to enable fast search operations.  $augp(x)$  stores the priority of the highest priority rule among the rules in the subtree

rooted at  $x$ . Thus, it can be recursively defined as follows:

$$augp(x) = \begin{cases} 0 & \text{if } x \text{ is nil} \\ pri(x) & \text{if } x \text{ is a leaf} \\ \max(augp(lchild(x)), augp(rchild(x)), pri(x)) & \text{otherwise} \end{cases}$$

If a BBST implementation, for example a 2-3 tree, is such that it stores data only in its leaf nodes,  $pri(x)$  is considered to be 0 and is ignored in the calculation of  $augp(x)$  for all internal nodes  $x$ . The augmented BBSTs for our running example are shown in Figure 2, with the  $augp$  field shown in bold and bigger font than the  $val$  and  $pri$  fields.

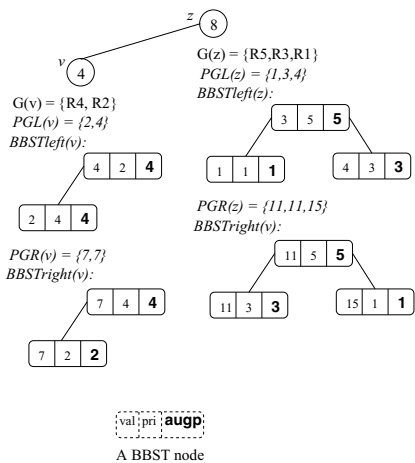


Fig. 2. Showing the BBSTs associated with each trie node.

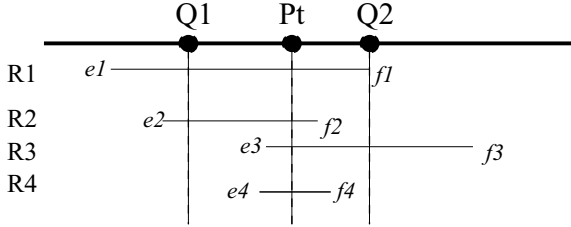
**Query Algorithm** Consider  $BBSTleft(v)$ , which stores the set of left endpoints,  $PGL(v)$ , for the trie node  $v$ . Since  $G(v)$  contains only those ranges that intersect  $Pt(v)$  and  $PGL(v)$  has the left endpoints of these ranges, all points in  $PGL(v)$  lie to the left of  $Pt(v)$  on the number line. Given a point  $Q$  representing an incoming packet, we define the following functions on the BBSTs at a trie node  $v$ :

1.  $gethpLeft(B, Q)$  — If the point  $Q$  is such that  $Q < Pt(v)$ , the function  $gethpLeft(B, Q)$  returns the priority of the highest priority range in  $G(v)$  that contains the point  $Q$ . Equivalently,  $gethpLeft()$  calculates  $\max_j \{pri(range(e_j)) \mid e_j \in PGL(v), e_j \leq Q\}$ . Note that  $e_j \leq Q$  if and only if  $range(e_j)$  contains  $Q$ . This function is performed by an algorithm that traverses the BBST  $B$  from its root to a leaf node. The algorithm looks at the current node being traversed in  $B$ , say  $v$ . Either  $Q < val(v)$  – in which case, the traversal descends to the left-child of  $v$ , or  $Q \geq val(v)$  – in which case, the highest priority

value found so far in the traversal is compared with  $\max(\text{augp}(v), \text{pri}(v))$  and updated if found smaller than this value. The traversal descends to the right child of  $v$  in this case. On reaching a null node, the maximum priority found by the traversal algorithm is the desired result.

2.  $\text{gethpRight}(B, Q)$  — If  $Q \geq \text{Pt}(v)$ , the function  $\text{gethpRight}(B, Q)$  returns  $\max_j \{\text{pri}(\text{range}(f_j)) \mid f_j \in \text{PGR}(v), f_j \geq Q\}$ . The algorithm is similar to that for  $\text{gethpLeft}(B, Q)$ .
3.  $\text{gethpTrieNode}(v, Q)$  — This function obtains the priority of the highest priority range in  $G(v)$  that contains the point  $Q$ . It first compares  $Q$  with  $\text{Pt}(v)$  and returns  $\text{gethpLeft}(\text{BBSTleft}(v), Q)$  if  $Q < \text{Pt}(v)$  or  $\text{gethpRight}(\text{BBSTright}(v), Q)$  otherwise.

The above functions are illustrated in Figure 3. Functions  $\text{gethpLeft}$  and  $\text{gethpRight}$  spend  $O(1)$  time at each node on the traversal path in the corresponding BBST. Since the depth of each BBST is  $O(\log |G(v)|)$ ,  $\text{gethpTrieNode}$  takes  $O(\log |G(v)|)$  or  $O(\log N)$  time for a classifier with  $N$  rules.



**Fig. 3.** The set of ranges in  $G(v)$  are shown in this figure. All ranges intersect  $\text{Pt} = \text{Pt}(v)$ . To calculate  $\text{gethp}(v, Q1)$ ; we first find that  $Q1 < \text{Pt}$ , and therefore calculate  $\text{gethpLeft}(\text{BBSTleft}(v), Q1)$ . This function considers ranges  $R_1$  and  $R_2$  returning the one with higher priority. Similarly for  $\text{gethp}(v, Q2)$ ,  $\text{gethpRight}(\text{BBSTright}(v), Q2)$  considers ranges  $R_1$  and  $R_3$ .

An incoming packet  $Q$  is classified using the function  $\text{gethpTrieNode}(v)$  as follows: we traverse the trie nodes according to the bits in  $Q$  in the usual manner. For each node  $v$  in the trie traversal path, we call the function  $\text{gethpTrieNode}(v)$  and calculate the maximum of all the returned values. This maximum value is then the highest priority rule matching  $Q$ . The query algorithm makes at most  $W$   $\text{gethpTrieNode}()$  calls and is therefore of complexity  $O(W \log N)$ .

The storage complexity of the *BoT* data structure is  $O(NW)$  because a BBST is a linear space data structure in the number of nodes. Hence total space consumption equals  $O(NW) + \sum_{v \in T} O(|G(v)|) = O(NW) + O(N) = O(NW)$ . Comparing the *HoT* and *BoT* query algorithms, we see that their worst case storage complexity is identical. In practice, however, we expect a classifier to require a smaller amount of storage in the *BoT* algorithm because a range is allocated to only one trie node as opposed to  $O(W)$  trie nodes in the *HoT* algorithm.

**Update algorithm.** We now describe the incremental update algorithms on a *BoT* data structure. We only describe the insertion algorithm – the deletion algorithm is similar and has the same complexity as the insertion algorithm. An incremental update needs to modify only the BBSTs of one trie node, the one that contains the rule to be updated. Modifying a BBST requires adding a new node to the BBST and appropriately updating the *augp* fields of all nodes. The addition of a new node has one non-trivial constraint – that the binary tree should be kept balanced. For concreteness, we describe the update algorithm under the assumption that a 2-3 tree is used for the implementation of BBSTs. The ideas are similar for other BBST implementations such as red-black trees.

A 2-3 tree stores the data only in its leaf nodes. Therefore, we assign a value of 0 to the *pri* field of each internal node and ignore this field in the calculation of the *augp* field for internal nodes. Every internal node of a 2-3 tree has either two or three children, and every path from the root node to a leaf node is of the same length. Insertion of a node with a new value,  $n$ , in a 2-3 tree is done in three phases:

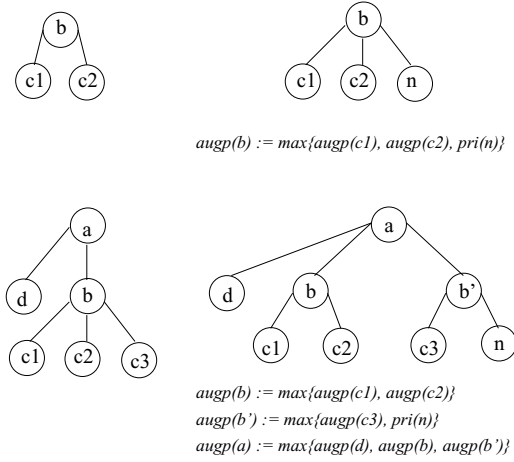
- (Phase 1) The algorithm descends the tree and adds the node  $n$  as a child of some node,  $b$ , in the order determined by the relative ordering of  $n$  and the values in the existing children nodes of  $b$ .
- (Phase 2) The addition of a new child to node  $b$  may result in  $b$  having four children. In order to restore the 2-3 property, we create a new node  $b'$ , make it the parent of two of the children nodes of  $b$ , and add  $b'$  as a child to the parent node of  $b$ . We recurse if the parent node of  $b$  now violates the 2-3 property. The recursive algorithm proceeds upward from  $b$  till it reaches the root of the tree.
- (Phase 3) The path from leaf  $n$  to the root is traced to update the value of the field *augp*.

The analysis of the insertion algorithm is simple. The depth of a 2-3 tree with  $N$  leaves is no more than  $\log N$ . Therefore phases 1 and 3 take  $O(\log N)$  time. Phase 2 also takes  $O(\log N)$  time because the recursive algorithm always moves up towards the root of the tree in one step, and  $O(1)$  work is performed at each step. Combining, the total complexity of the insertion algorithm is  $O(\log N)$ . The deletion algorithm is similar and therefore has the same complexity. In summary, incremental updates take  $O(\log N)$  time on a *BoT* data structure.

## 4.2 Multi-dimensional Classifiers

The *BoT* data structure can be extended to multiple dimensions in a manner similar to the *HoT* data structure by using multi-level tries, one level for each dimension except the last, on which a *BoT* is built. For example, in the *BoT* for a two dimensional classifier, the ranges of the rules in one dimension are allocated to  $O(W)$  nodes of the first level trie and a one-dimensional *BoT* as explained above will be built on the nodes of this first level trie using the ranges of the rules in the second dimension. The total space consumed is then  $O(NW^d)$





**Fig. 4.** Showing the different cases of an update operation.

with query and update time complexities being  $O(W^d \log N)$  and  $O(W^{d-1} \log N)$  respectively.

## 5 Conclusions and Open Problems

This paper presented two dynamic algorithms for multi-dimensional packet classification. The complexities of these algorithms are as shown in Table 1 where  $W$  (the number of bits used in each dimension) has been approximated by  $\log N$ . The choice of algorithm in a practical application will be determined by the query and update time requirements.

There have been a number of proposed heuristics for packet classification, each with its pros and cons. In general, these algorithms exploit structure present in existing classifiers. Other non-heuristic algorithms trade off update time for reduced query time, or do not generalize to multiple dimensions. While these algorithms perform well today on relatively small, well-understood classifiers and on classifiers that change infrequently, they will perform less well in future when classifiers change rapidly (e.g., in a router providing service differentiation), or when classifiers evolve to have different structure.

As a first step towards future requirements, the *Hot* and *BoT* algorithms do not trade off among the metrics of query time, update time and space requirements. While still lacking in some implementation details, both algorithms have reasonable worst-case bounds in all three metrics for general multi-dimensional classifiers.

Finally, we pose two questions that are natural consequences of this paper:

1. What are non-trivial lower bounds to the query time in the  $d$ -dimensional packet classification problem, both static and dynamic version, in a given amount of space?

2. Can we improve upon *both* the search and update times of Table 1 in the same or better space complexity?

## Acknowledgments

We wish to gratefully acknowledge Michael Lamoureux for useful discussions and for giving access to his bibliography on multi-dimensional search. We also wish to acknowledge Leo Guibas for useful discussions and Youngmi Joo for useful comments on a draft of this paper.

## References

1. Cisco Systems white paper, "Advanced qos services for the intelligent internet," [http://www.cisco.com/warp/public/cc/cisco/mkt/ios/qos/tech/qos\\_wp.htm](http://www.cisco.com/warp/public/cc/cisco/mkt/ios/qos/tech/qos_wp.htm).
2. T. V. Lakshman and D. Stiliadis, "High-speed policy-based packet forwarding using efficient multi-dimensional range matching," in *Proceedings of ACM SIGCOMM*, Sept. 1998, pp. 191–202.
3. V. Srinivasan, G. Varghese, S. Suri, and M. Waldvogel, "Scalable level 4 switching and fast firewall processing," in *Proceedings of ACM SIGCOMM*, Sept. 1998, pp. 203–214.
4. P. Gupta and N. McKeown, "Packet classification on multiple fields," in *Proceedings of ACM SIGCOMM*, Sept. 1999, pp. 147–160.
5. D. Decasper, Z. Dittia, G. Parulkar, and B. Plattner, "Router plugins: A software architecture for next generation routers," in *Proceedings of ACM SIGCOMM*, Sept. 1998, pp. 229–240.
6. P. Gupta and N. McKeown, "Packet classification using hierarchical intelligent cuttings," *Hot Interconnects VII*, Aug. 1999.
7. V. Srinivasan, G. Varghese, and S. Suri, "Fast packet classification using tuple space search," in *Proceedings of ACM SIGCOMM*, Sept. 1999, pp. 135–146.
8. M. M. Buddhikot, S. Suri, and M. Waldvogel, "Space decomposition techniques for fast layer-4 switching," *Protocols for High Speed Networks*, vol. 66, no. 6, pp. 277–283, Aug. 1999.
9. A. Feldmann and S. Muthukrishnan, "Tradeoffs for Packet Classification," in *Proceedings of INFOCOM*, Mar. 2000.
10. W. Doeringer, G. Karjoth, and M. Nassehi, "Routing on longest-matching prefixes," *IEEE/ACM Transactions on Networking*, vol. 4, no. 1, pp. 86–97, Feb. 1996.
11. K. Mehlhorn and S. Naher, "Dynamic fractional cascading," *Algorithmica*, vol. 5, pp. 215–241, 1990.
12. L. J. Guibas, "Private communication," .
13. T. H. Cormen, C. E. Leiserson, and R. L. Rivest, *Introduction to Algorithms*, McGraw-Hill, 1990.
14. A. V. Aho, J. E. Hopcroft, and J. D. Ullman, *The Design and Analysis of Computer Algorithms*, Addison-Wesley, 1974.

# Intelligent Traffic Conditioners for Assured Forwarding Based Differentiated Services Networks

Biswajit Nandy, Nabil Seddigh, Peter Piedad, and Jeremy Ethridge

OpenIP Group, Nortel Networks, Ottawa, Canada  
{bnandy, nseddigh, ppieda, jethridg}@nortelnetworks.com

**Abstract.** Issues related to bandwidth assurance in Assured Forwarding based Differentiated Services (Diffserv) networks have been discussed in recent research papers [7][8][11]. Some of the factors that can bias bandwidth assurance are Round Trip Time (RTT), UDP/TCP interaction and different target rates. The bias due to these factors needs to be mitigated before bandwidth assurance for a paying customer can be articulated in Service Level Agreements (SLAs). This paper proposes intelligent traffic conditioning approaches at the edge of the network to mitigate the effect of Round Trip Time, UDP/TCP interactions, and different target rates. The simulation results show a significant improvement in bandwidth assurance with intelligent traffic conditioning. The limitation of the proposed solutions is that they require communication between edge devices. In addition, these solutions are not applicable for a one-to-any network topology.

## 1 Introduction

The Differentiated Services (Diffserv) architecture [2] has recently become the preferred method to address QoS issues in IP networks. This packet marking based approach to IP-QoS is attractive due to its simplicity and ability to scale. An end-to-end differentiated service is obtained by concatenation of per-domain services and Service Level Agreements (SLAs) between adjoining domains along the path that the traffic crosses in going from source to destination. Per domain services are realized by traffic conditioning at the edge and simple differentiated forwarding mechanisms at the core of the network. Two forwarding mechanisms recently standardized by the IETF are the Expedited Forwarding (EF) [6] and Assured Forwarding (AF) [5] Per Hop Behaviors (PHB).

The basis of the AF PHB is differentiated dropping of packets during congestion at the router. The differentiated dropping is achieved via “RED-like” [1] Active Queue Management (AQM) techniques. The AF PHB RFC specifies four classes and three levels of drop precedence per class. AF is an extension of the RIO [3] scheme, which uses a single FIFO queue and two levels of drop precedence.

To build an end to end service with AF, subscribed traffic profiles for customers are maintained at the traffic conditioning nodes at the edge of the network. The aggregated traffic is monitored and packets are marked at the traffic conditioner.

When the measured traffic exceeds the committed target rate, the packets are marked with higher drop precedence (DP1) otherwise packets are marked with lower drop precedence (DP0). If the measured traffic exceeds the peak target rate, the packets are marked with highest drop precedence (DP2). At the core of the network, at the time of congestion, the packets with DP1 marking have higher probability of being dropped than packets with DP0 marking. Similarly, packets with DP2 marking have higher probability of being dropped than packets with DP0 and DP1 marking. The different drop probabilities are achieved by maintaining three different sets of RED parameters – one for each of the drop precedence markings

Although the IETF Diffserv Working Group has finalized the basic building blocks for Diffserv, we argue that there are many open issues in understanding and evaluating, the kinds of end-to-end services that could be created for an end user using the AF PHB. Various issues with bandwidth assurance in a Diffserv network have been reported in recent research papers [4][11]. A number of these issues need to be resolved before quantitative assurances of some form can be specified in SLA contracts.

The key contribution of this paper is the proposal of intelligent traffic conditioners to improve the bandwidth assurance for AF-based services and mitigate the effects of various factors in biasing the achieved bandwidth. A RTT-Aware Marker based on the TSW [3] is developed to reduce the effects of RTT in determining the achieved bandwidth for TCP flows. Extensive study is performed to consider whether UDP/TCP fairness issues can be solved via intelligent mapping of TCP and UDP to different drop precedence or AF classes. Finally, two TargetRate-Aware Markers are presented with the objective of distributing excess bandwidth in proportion to the target rates.

The rest of this paper is organized as follows. Related work is examined in the next Section. Section 3 describes the topology of the test network and various simulation parameters. Section 4 presents the solution to mitigate the impact of RTT. TCP/UDP interaction issues are addressed in Section 5. An algorithm for excess bandwidth distribution in proportion to target rates is discussed in Section 6. Section 7 provides an analysis, discussion and evaluation of the proposed solutions. Section 8 contains concluding remarks and points to areas of future work.

## 2 Related Work

Clark and Fang [3] reported the initial simulation study on a differentiated drop scheme. The paper introduced RIO (RED with In/Out) and a remarking policer that utilized an average time sliding window (TSW) rate estimator and intelligent marker. The main contribution of that work was to show that source target rates could be assured in a simple capacity allocated network that relies on statistical multiplexing. Ibanez and Nichols [4] via simulation studies, showed that RTT, target rate, TCP/UDP interactions are key factors in the throughput of flows that obtain an Assured Service using a RIO-like scheme. Their main conclusion is that such an Assured Service “cannot offer a quantifiable service to TCP traffic”. Seddigh, Nandy and Piedad [11] have confirmed with detailed experimental study that the above mentioned factors are

critical for biasing distribution of excess bandwidth in an over-provisioned network. In addition, it has been shown [11] that the number of micro-flows in an aggregate and packet sizes play a key role in determining the bandwidth achieved in over-provisioned networks.

Recently, various researchers [7][12][14] have reported new approaches to mitigate the biasing effects of some of the factors outlined in [4] and [11]. Lin, Zheng and Hou [7] have proposed an enhanced TSW profiler and two enhanced RIO queue management algorithms. The simulation results show that the combination of enhanced algorithms improves the throughput and fairness requirements especially with different target rates, RTTs and co-existing UDP flows. However, the proposed solutions may not be scaleable due to the usage of state information at the core of the network.

Yeom and Reddy [12] have suggested an algorithm that improves fairness for the case where the individual flows in an aggregate have different RTTs. The proposed algorithm maintains per flow information at the edge of the network. Kim [14] proposes a token allocation scheme to distribute tokens to individual flows originating from the same subscriber network. The paper claims that using this approach, fairness in TCP and UDP interaction and fairness between TCP connections with different RTTs can be achieved. The details of the algorithm are not clearly reported in the IETF draft [14].

3 Simulation Detail

The studies in this paper were performed using the ns-2 simulator [15]. The simulator was enhanced to include networking elements with Diffserv edge and core device functionality as specified in [2].

The network topology used in the experiments can be seen in Figure 1. The setup consisted of three network edge devices E1, E2, E3 and one core device C1. Each edge device is connected to an end host or traffic source. The TCP flows generated were all long lasting. Experiments with RTT-Aware Traffic Conditioner are performed with the network topology shown in Figure 1. The topology of the network for the experiments with TCP/UDP interaction is an extension of Figure 1. Six edges are connected to six separate traffic sources. The Target Rate-Aware Traffic Conditioner also utilizes the same topology with six edges and sources.. The bottleneck link is between core device C1 and edge device E3.

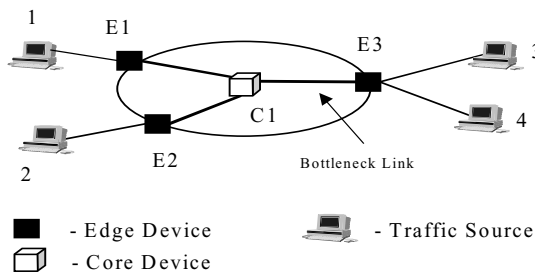


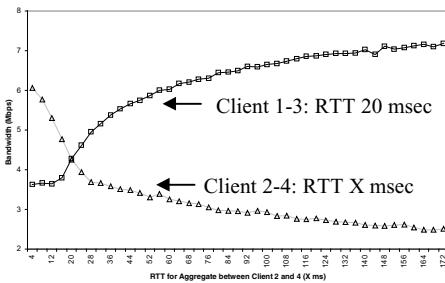
Fig. 1. Simulation Testbed

The Edge devices in the testbed classify packets based on source and destination IP addresses. The policer utilizes the Time Sliding Window (TSW) tagger [3]. This is referred to as the Standard Traffic Conditioner (TC). The core device implements the AF PHB using the three-colour version of RIO[10]. Unless otherwise stated, the experiments in the paper use RED parameters with  $w_q=0.002$ . RED  $\min_{th}$ ,  $\max_{th}$  and  $\max_p$  thresholds are  $DP0=\{40,55,0.02\}$ ;  $DP1=\{25,40,0.05\}$ ;  $DP2=\{10,25,0.1\}$

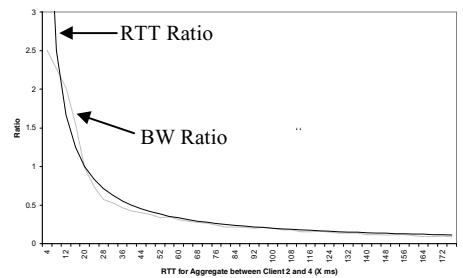
## 4 Mitigating the Impact of Round Trip Time

Studies have shown that in Best Effort networks [9], the bandwidth achieved by TCP flows is a function of the Round Trip Time (RTT). This is due to TCP's use of a self-clocked sliding window based mechanism. Recent studies [11] have shown that flows with different RTTs, despite having identical target rates, will get different shares of the bandwidth. For over-provisioned networks, the flows will mostly achieve their target rate irrespective of their RTTs[11][12]. This is because DP0 traffic is protected and DP1 traffic will be dropped before any DP0 packets are dropped. However, there will be an unfair sharing of the excess bandwidth in favor of those target aggregates with lower RTTs. In the under-provisioned case, neither of the aggregated flows will achieve their target. However, the flows with high RTT will be further away from the target than the flows with low RTT.

An initial simulation is performed to show the impact of RTT on bandwidth and develop the basis for the RTT-Aware Traffic Conditioner. Two traffic aggregates are generated. Each aggregate has target rate of 2 Mbps. Each aggregate (between client 1 and 3; and between client 2 and 4) has six TCP flows. This profile results in a total allocated bandwidth of 4 Mbps, which is 40% of the bandwidth at the bottleneck link. The transmission delay between edges E1 and E3 ( $RTT_{13}$ ) is kept at 20 ms while  $RTT_{24}$  (between client 2 and 4) is varied from 1 to 200 ms.



**Fig. 2.** Achieved BW Using Standard TC



**Fig. 3.** BW and RTT Ratio Using Standard TC

Figure 2 shows the total bandwidth achieved by each aggregate. The Figure shows that as the RTT between client 2 and 4 is increased, the share of bandwidth of the aggregate decreases. The result reflects the steady state TCP behavior as reported by Mathis et al. [9]. Equation (1) shows that the BW is inversely proportional to RTT.

$$BW \propto \frac{MSS}{RTT * \sqrt{p}}, \text{ where MSS is the segment size and } p \text{ is pkt drop probability} \quad (1)$$

As the drop rate and MSS are same for both traffic aggregates, from Equation (1) the

$$\text{BW ratios can be represented as: } \frac{BW_{24}}{BW_{13}} = \frac{RTT_{13}}{RTT_{24}} \quad (2)$$

Figure 3 plots the ratio of RTTs and bandwidth from the simulation results. It shows that the ratio of the two RTTs is identical to the inverse ratio of the measured TCP aggregate bandwidth between clients 1-3 and 2-4 thus verifying equation 2. Equation 1 forms the basis of the RTT-Aware traffic conditioner.

### RTT-Aware Traffic Conditioning

Various approaches are possible to address the impact of RTT on TCP throughput. One approach is to modify the TCP windowing mechanism at the end host and make it RTT aware. A second method is to use the knowledge of RTT to affect dropping at the congested core devices. A third alternative is to introduce a mechanism at the edge of the network to handle the impact of RTT on throughput. We have taken the third approach.

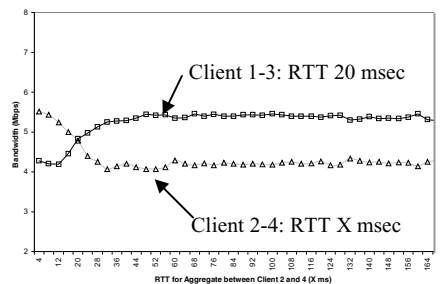
From equation 1 it is seen that if the packet drop rate can be adjusted in relation to RTT, the acquired bandwidth for the aggregate can be made insensitive to RTT. This is the basis of the RTT-aware Traffic conditioning algorithm. The aggregates with high RTT take longer to ramp up after a packet drop occurs. Thus the achieved average bandwidth for high RTT aggregates are lower. Protecting higher amount of traffic for long RTT aggregates can compensate the loss in bandwidth. Our approach increases the amount of in-profile traffic for high RTT aggregates in a proportional manner.

```

If (measuredRate <= TargetRate)
    /* i.e., IN-profile */
    Map Packets to "dp0"
Else /* i.e., OUT-of-profile */
    Map Packets to "dp0" with probability (1-p)
    Map Packets to "dp1" with probability p
Where:  $p = q * r$ 
 $q = \frac{(MeasuredRate - TargetRate)}{MeasuredRate}$ 
 $r = \left( \frac{\min RTT}{aggregateRTT} \right)^2$ 

```

**Fig. 4.** RTT-Aware TC Algorithm



**Fig. 5.** Achieved BW using RTT-Aware TC

Figure 4 outlines the RTT-Aware packet marking algorithm. The algorithm is an extension of the TSW marker[3]. A derivation of the algorithm can be found in the detailed version of this paper[13]. As long as the measured sending rate remains below the target rate the packets are marked with DP0. Beyond the target rate, packets are marked DP1 with probability  $p$  and DP0 with probability  $(1-p)$ . The probability  $p$  is calculated using knowledge of the traffic stream's measured RTT relative to the minimum RTT (minRTT) in the DS domain. For traffic streams with lower RTT, packets beyond the target rate will get marked to DP1 with higher probability. At the time of congestion, more packets with DP1 marking will be susceptible to dropping, thus adjusting the achieved bandwidth. Three assumptions for this scheme are: (a) all the flows in the aggregate have the same RTT i.e., source and destination points are same. (b) minRTT of the network is known to all the edge devices. (c) RTT for the aggregate flow is known at the edge of the network.

We repeat the same experiment for which the result was shown in Figure 2; except the RTT-Aware TC is used instead of standard TC. Figure 5 shows the results. Comparing Figure 5 with Figure 2, we observe that the impact of RTT has been significantly mitigated. The two aggregates achieve a similar share of the excess bandwidth.

One major assumption for the RTT-Aware TC is that the TCP flows are operating in congestion avoidance state. Equation (1) is not representative of bandwidth achieved if flows are in slow start. With a large number of flows in an aggregate and inappropriate setting of RED parameters, many flows can timeout and enter slow-start repeatedly. In such a case, it has been observed that the RTT-Aware TC is less effective in mitigating the impact of RTT in biasing bandwidth distribution. The next sub-section discusses the issues with large number of flows and studies the applicability of the proposed RTT-Aware marking algorithm.

## 5 Addressing Fairness Issues with TCP/UDP Interactions

A paying Diffserv customer will inject both TCP and UDP traffic to the Diffserv network. The interaction between TCP and UDP may cause the unresponsive UDP traffic to impact the TCP traffic in an adverse manner. There clearly, is a need to ensure that responsive TCP flows are protected from non-responsive UDP flows, but at the same time protect certain UDP flows which require the same fair treatment as TCP due to multimedia demands. Moreover, we argue it is the Diffserv customer who should decide the importance of the payload assuming the network is capable of handling both TCP and UDP traffic in a fair manner. We suggest three fairness criteria for TCP and UDP traffic are:

1. In an over-provisioned network, both UDP and TCP target rates should be achieved.
2. In an over-provisioned network, UDP and TCP packets should have a reasonable share of the excess bandwidth. Neither TCP nor UDP should be denied access to the excess bandwidth.
3. In an under-provisioned network, TCP and UDP flows should experience degradation in proportion to their target bandwidth.



There are two possible approaches to solve the fairness issues: (a) Mapping TCP and UDP to different drop precedence of the same AF class, (b) Mapping TCP and UDP to different AF class queues.

Experiments are performed with two UDP sources with target rates of 1Mbps and sending rate of 6 Mbps each - CBR. High sending rates of UDP flows are chosen so that the impact of UDP on TCP can be easily evaluated. Four TCP aggregates are generated with each aggregate consisting of 32 flows. The target rate of each TCP aggregate is varied from 0.5Mbps to 3Mbps. Thus the total target rates of UDP and TCP aggregates are varied from 4Mbps to 14Mbps so that bandwidth allocation at the core of the network changes from over-provisioned state (40% allocated capacity) to under-provisioned state (140% allocated capacity).

Mapping TCP and UDP to Different Drop Precedence

Drop precedence mapping scheme is one way to ensure fairness for both TCP and UDP. This study attempts to evaluate various options of mapping TCP and UDP to different drop precedence based on the matrix in Table 1. In all scenarios, TCP traffic within the target bandwidth (“IN-Profile”) is assigned to DP0. In scenarios 1 and 6, UDP in-profile traffic is also assigned to DP0. UDP in-profile traffic assignment to DP1 (in scenarios 2, 3 and 4) and to DP2 (in scenario 5) are also considered. The experiments are performed with intelligent TC to perform appropriate mapping at the edge of the network.

Table 1. Possibilities for Mapping TCP and UDP to different drop precedences

Scenarios	1	2	3	4	5	6
TCP-IN Profile	DP0	DP0	DP0	DP0	DP0	DP0
TCP-OUT-of Profile	DP1	DP1	DP1	DP2	DP1	DP1
UDP-IN Profile	DP0	DP1	DP1	DP1	DP2	DP0
UDP-OUT-of Profile	DP1	DP1*	DP2	DP2	DP2*	DP2

\* No distinction is made between UDP-IN and UDP-OUT packets

In Scenario 1, both UDP and TCP in-profile packets are mapped to DP0 and out-of-profile packets are mapped to DP1. Both TCP and UDP flows achieve their target bandwidth in an over-provisioned network (Figure 6). The UDP flows get most of the share of the excess bandwidth. As the network approaches an under-provisioned state, the TCP flows suffer more degradation than the UDP flows. This is due to identical mapping of TCP and UDP out-of-profile traffic.

It is observed that Scenario 2 to 5 cannot assure UDP target bandwidth. This is due to the allocation of UDP in-profile traffic to DP1 or DP2. Thus, UDP in-profile traffic is dependent on DP0 TCP traffic. Total in-profile TCP traffic determines if the target rate of UDP can be achieved or not. In other word, the buffer occupancy of UDP in-profile traffic is dependent on the TCP in-profile traffic in DP0. Sharing of excess bandwidth is dependent on the assigned drop precedence of TCP and UDP. Similar argument is true for under-provisioned scenario.

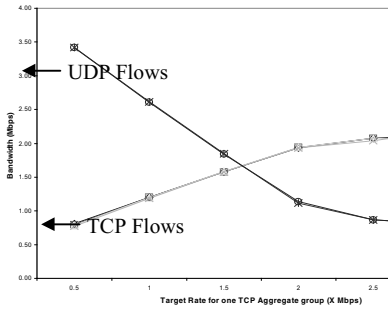


Fig. 6. Scenario 1: Achieved BW

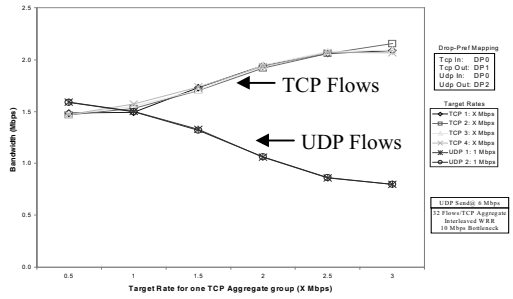


Fig. 7. Scenario 6: Achieved BW

In Scenario 6 both TCP and UDP in-profile packets are mapped to DP0. However, TCP out-of-profile packets are mapped to DP1 while UDP out-of-profile packets are mapped to DP2. The results are shown in Figure 7. In the over-provisioned case, both TCP and UDP achieve their target bandwidth. However, TCP obtains a greater share of the excess bandwidth than UDP. In an under-provisioned network, the TCP flows experience greater degradation from their target bandwidth, than the UDP flows.

The results show that the target bandwidth for TCP and UDP flows can be achieved by protecting the in-profile traffic and mapping it to DP0. For an over-provisioned network, the manner in which the excess bandwidth is shared (i.e., fairness criteria 2) remains dependent on the drop precedence assignment of TCP and UDP out-of-profile packets. In an under-provisioned network (i.e., fairness criteria 3), isolation of TCP and UDP in-profile traffic is necessary. Mapping both UDP and TCP in-profile to the same drop precedence (i.e., scenario 1 and 6) results in unfairness to TCP as it experiences degradation from its target bandwidth in comparison to UDP.

### Mapping TCP and UDP to Different AF Class Queues

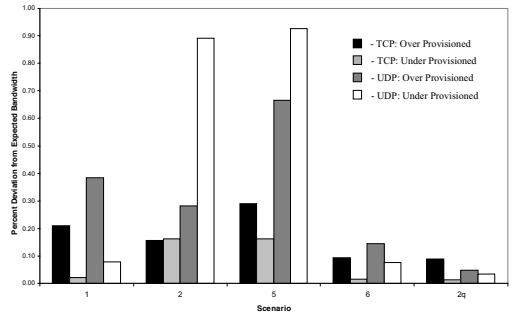
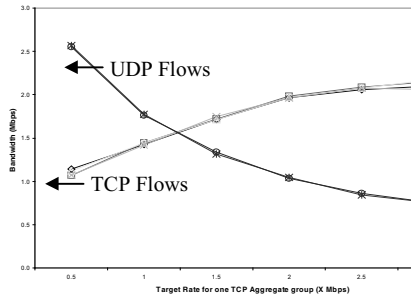
Another way to achieve fairness is to completely isolate the TCP and UDP traffic in two separate AF class queues at the core of the network. At the edge of the network, the intelligent TC marks the TCP and UDP packets to different AF classes. A weighted scheduling scheme is used at the core to enforce fairness among TCP and UDP flow aggregates. If the weights of the scheduling class queues are distributed in proportion to the TCP and UDP target rates, the fairness criteria can be satisfied. The weights for the queues can be selected using following method:

$$W_{TCP} = \left( \frac{\sum_{i=1}^n R_{TCP}^i}{\left( \sum_{i=1}^n R_{TCP}^i + \sum_{i=1}^m R_{UDP}^i \right)} \right) \quad W_{UDP} = \left( \frac{\sum_{i=1}^m R_{UDP}^i}{\left( \sum_{i=1}^n R_{TCP}^i + \sum_{i=1}^m R_{UDP}^i \right)} \right)$$

- where  $R_{TCP}^i$ : Target Rate for TCP aggregate i  
 $R_{UDP}^i$ : Target Rate for UDP aggregate i  
 $W_{UDP}$ : Weight for UDP Scheduling Queue  
 $W_{TCP}$ : Weight for TCP Scheduling Queue

The above equations set the weight assuming that UDP aggregates are sending packets at rate equivalent to or greater than their target rates. TCP traffic and UDP traffic are mapped to different AF classes. IN-profile traffic is mapped to DP0 and OUT-of-profile traffic is mapped to DP1. A weighted round robin scheduler is used to schedule packets between two queues at the core of the network. The traffic mix is the same as used for results of Figure 6 and Figure 7.

The results are depicted in Figure 8. It is observed that all the three fairness criteria are satisfied. Both TCP and UDP achieve their target rates. In the over-provisioned case, TCP and UDP obtain a reasonably fair share of the excess bandwidth. In the under-provisioned case, the aggregated bandwidth for TCP and UDP degrades proportionally.



**Fig. 8.** Scenario: Two Queue - Achieved BW

**Fig. 9.** Deviation from Expected Bandwidth

## Fairness Analysis

To further compare the results against the original fairness criteria, quantitative analysis is performed. The analysis compared the expected versus the actual bandwidth obtained for each of the seven scenarios. The percentage deviation from expected share of bandwidth is calculated. These values give two sets of averages, one for the four TCP aggregates and one for the two UDP aggregates. To facilitate detailed analysis, we distinguish between under-provisioned and over-provisioned cases. For the over-provisioned calculation, we use points for case of TCP aggregate target rate with values 0.5, 1, 1.5; for under-provisioned, we use 2, 2.5 and 3.

Equation (3) calculates the expected fair share of the bandwidth for UDP customers. The maximum sending Rate of UDP is considered by taking the minimum between the UDP stream maximum sending rate and the fair share bandwidth.

$$\text{Expected UDP Agg BW} = \text{Min} \left\{ \left( \frac{R_{UDP}^i}{\left( \sum_{i=1}^n R_{TCP}^i + \sum_{j=1}^m R_{UDP}^j \right)} \right) \times BW_{link}, S_{udp}^i \right\} \quad (3)$$

$$R_{udp}^i = \text{Target rate (Mbps) for UDP customer } j$$

$$\begin{aligned}
R_{tcp}^j &= \text{Target rate (Mbps) for TCP customer } i \\
BW_{link} &= \text{Link bandwidth (Mbps)} \\
S_{udp}^i &= \text{Maximum sending rate (Mbps) for UDP customer } i
\end{aligned}$$

Equation (5) determines the expected fair share of the bandwidth for TCP aggregates. Unused bandwidth (equation 4) from the UDP aggregate(s) is divided between the TCP aggregates, proportional to their target rate. UDP aggregates have unused bandwidth when their sending rate is below the target rate.

Total Unused UDP BW is given by:

$$U_{UDP} = \text{Max} \left\{ \left[ \sum_{k=1}^m \left( \frac{R_{UDP}^k}{\left( \sum_{i=1}^n R_{TCP}^i + \sum_{j=1}^m R_{UDP}^j \right)} \right) \times BW_{link} - \sum_{k=1}^m S_{udp}^k \right], 0 \right\} \quad (4)$$

$$\text{Expected TCP Customer BW} = \left( \frac{R_{TCP}^i}{\left( \sum_{i=1}^n R_{TCP}^i + \sum_{i=1}^m R_{UDP}^i \right)} \right) \times (BW_{link} + U_{udp}) \quad (5)$$

$$\text{Deviation from Expected BW} = \left| \frac{\text{ExpectedBandwidth} - \text{MeasuredBandwidth}}{\text{ExpectedBandwidth}} \right| \quad (6)$$

The graph in Figure 9 illustrates the results of the quantitative analysis for scenarios 1, 2, 5, 6 and 2q. Results for scenario 3 and 4 are similar to 2 and 5 and omitted due to space constraints. From the graph we can see that the test with two class queues had the least deviation from expected BW. Deviation in Scenario 6 is also comparable to the test with two class queues. Scenario 1 has high deviation for excess BW (for both TCP and UDP). UDP performs poorly for under provisioned cases in Scenario 2-5.

## 6 Excess BW Distribution for Aggregates with Different Target Rates

In a Diffserv network, different customers will contract different target rates. Recent research has shown that in an over-provisioned network, with standard TC, there is an almost even distribution of excess bandwidth irrespective of the target rate[11]. This may not be an acceptable solution, as the high paying customer with higher target rate will expect a higher share of the excess bandwidth. Further discussion on the merit of equal versus proportional distribution of excess bandwidth can be found in section 7. This work assumes proportional distribution is desirable.

This section describes and evaluates two intelligent traffic conditioners developed to address the issue of proportional distribution of excess bandwidth. The first solution uses DP0 and DP1 and is referred to as Target Aware TC with two drop precedence (TATC-2DP). The TATC-2DP approach is similar to the RTT-Aware TC. The excess out-of-profile traffic is allocated back to in-profile in proportion to the target rates. This will lead to higher assured bandwidth for aggregates with high target rate. The algorithm in Figure 10 outlines the TATC-2DP marking scheme for the traffic conditioner.

The second solution uses all three drop precedence and is called TATC-3DP. In this scheme, the excess bandwidth is divided between DP1 and DP2 in proportion to the target rate. The algorithm for the TATC-3D is captured in Figure 11.

```

If (measuredRate <= TargetRate)
    /* i.e., IN-profile */
    Map Packets to "dp0"
Else /* i.e., OUT-of-profile */
    Map Packets to "dp0" with probability (1-p);
    Map Packets to "dp1" with probability p;

Where:  $p = q * r$  :

$$q = \frac{(MeasuredRate - TargetRate)}{MeasuredRate}$$


$$r = \left( \frac{\min TargetRate}{aggregateTargetRates} \right)^2$$


```

**Fig. 10.** The TATC-2DP Algorithm

```

If (measuredRate <= TargetRate) /*IN-profile */
    Map Packets to "dp0"
Else /* i.e., OUT-of-profile */
    Map Packets to "dp0" with probability (1-p);
    If (packet is not marked "dp0")
        Map Packets to "dp1" with probability 1-q;
        Map Packets to "dp2" with probability q;

Where: p and q:

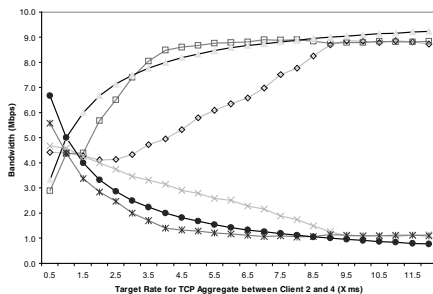
$$p = \frac{(MeasuredRate - TargetRate)}{MeasuredRate}$$


$$q = \left( \frac{\min TargetRate}{AggregateTargetRate} \right)$$

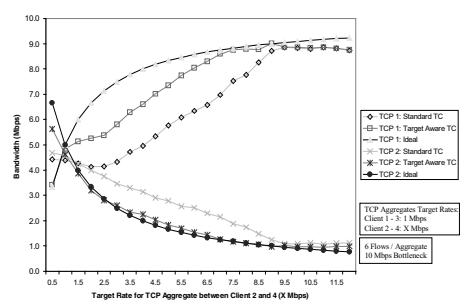

```

**Fig. 11.** The TATC-3DP Algorithm

We perform the same set of experiments using both the TATC-2DP and the TATC-3DP. The first experiment is performed with two sets of aggregates from clients 1 to 3 and 2 to 4 respectively. Each aggregate consists of six TCP flows. One aggregate has a target rate of 1Mbps and the other aggregate has a target rate that is varied between 0.5 to 11.5 Mbps; thus creating a capacity allocation from 15% to 120% at the bottleneck link.



**Fig. 12.** Achieved BW using TATC-2DP



**Fig. 13.** Achieved BW using TATC-3DP

Figure 12 shows the results of the experiment with standard TC and Target-Aware TC when the TATC-2DP algorithm is used. The expected bandwidth is also plotted. It is observed that there is a gap in the expected and achieved bandwidth when standard TC is used. The excess bandwidth is not proportionally distributed as would be desired by a customer. Instead, we see an almost even distribution of the excess bandwidth between two sets of competing flows. When the Target-Aware TC is used, the achieved bandwidth is closer to the expected bandwidth for both the flow aggregates. Similar results are shown in Figure 13 for the case when the TATC-3DP algorithm is used.

In another experiment, six different flow aggregates with different target rates are pushed through bottleneck links of 45 Mbps and 22 Mbps. The total allocated target rate constitutes 40 % and 80% of the bottleneck link capacity respectively. The experiment is performed with standard TC and TATC-3DP. Figure 14 shows the achieved bandwidth for all aggregates in case of the 45 Mbps bottleneck link. Figure 15 shows the achieved bandwidth for all the aggregates in case of 22 Mbps bottleneck link. It is seen that improvement in bandwidth allocation is significant for heavily over-provisioned network. The experiment was repeated for the TATC-2DP and the results closely resemble those in Figures 14 and 15.

Table 2 reflects the extent to which the different Target-Aware TCs are able to achieve the expected bandwidth based on proportional distribution of the excess. The table shows the average deviation from expected value achieved by each traffic conditioner in each of the three experiments performed in this section.

For all three experiments, it can be concluded that the standard TC has a higher percentage deviation from expected results than either of the TATC algorithms. The performance of TATC-2DP and TATC-3DP are comparable .

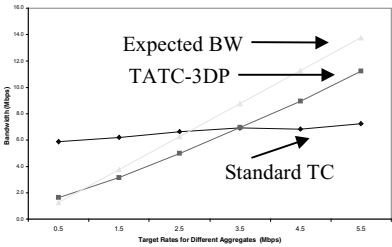


Fig. 14. TATC-3DP: 40% Capacity Allocation

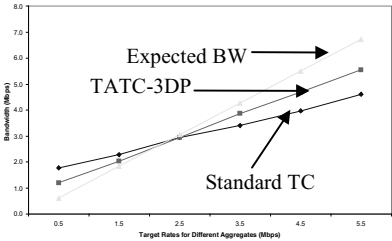


Fig. 15. TATC-3DP: 80% Capacity Allocation

Table 2. Percentage Average Deviation from Expected Results.

TC	Figures 12 & 13	Figures 14 & 15	
		80 % Provisioned	40 % Provisioned
Standard	0.32	0.50	0.91
TATC-2DP	0.11	0.23	0.19
TATC-3DP	0.08	0.25	0.21

## 7 Discussion

The previous sections have presented various methods for ensuring a fairer distribution of bandwidth for flows in an AF-based Diffserv network. In this section, we evaluate the applicability of the proposed solutions and identify the limitations.

### RTT-Aware TC

The RTT-Aware TC has the following requirements. Firstly, it is applicable for traffic streams where all flows in the aggregate have the same RTT. Secondly, it requires the edge devices to determine the RTT of aggregates passing through it. One possible way to do this is to consider a single flow as representative of the aggregate. The edge device can perform RTT measurement of the aggregate traffic at the edge of the network. This will require per-flow state monitoring of data packets and observing the return of corresponding ACKs in the reverse direction. Such a scheme assumes that the delay from the edge to the host is minimal.

The third requirement is to determine the minimum RTT for aggregates in the network. Two approaches are possible. If queueing delay at core devices is minimal then for pre-configured point-to-point connections, the RTT can be estimated based on transmission delay of intermediate links. If however, queueing delay is a major component in the RTT then the RTT needs to be dynamically measured. Thus, to determine the minRTT, the edge nodes need to exchange RTT information cooperatively.

### TCP/UDP Interaction

The TCP/UDP studies (Figure 9) showed that using drop precedence mapping, certain level of fairness can be achieved. Mapping TCP and UDP in-profile traffic to DP0 helps to achieve the target bandwidth. However, mapping of TCP and UDP out-of-profile traffic to different drop precedence is necessary to handle the bandwidth distribution at over-provisioned and under-provisioned states. Scenario 6 satisfies the required mapping and it is reflected in low percentage deviation in fairness index (Figure 9). As shown in Figure 9, use of two queues to isolate TCP and UDP traffic provides the optimum solution. However, the approach has a possible drawback due to the necessity of knowing the fraction of TCP and UDP target rates at the core of the network. This can be handled by the use of bandwidth broker - to communicate target rates - or by pre-allocating weights for each queue based on an estimate of UDP and TCP traffic.

### Target-Aware TC

It is debatable whether the excess bandwidth in an over-provisioned network should be divided among aggregates in proportion to the subscribed target rates or should be divided equally. This is a business decision that shouldn't be influenced by technical limitations. Should providers wish to offer a proportional distribution of the excess, they should have the building blocks at their disposal to do so. The TATC-2DP and TATC-3DP are two examples of such building blocks.

Although the performance results of the TATC-2DP and the TATC-3DP are comparable, there are practical issues to consider when evaluating a Target-Aware TC. The TATC-2DP will increase the amount of in-profile traffic in the network. This makes traffic engineering more difficult because the total in-profile traffic cannot be estimated from the subscribed total target-rates. On the contrary, the TATC-3DP has in-profile traffic that is consistent with the target rates since excess traffic is partitioned between DP1 and DP2.

Both the TATC schemes require knowledge of the minimum Target Rate in the network. This is not as difficult to obtain as the minimum RTT in the network. Target Rates are typically static and don't change as often as RTT. Thus, the minimum Target Rate can be periodically determined via the existing policy management framework and communicated to the edge devices using a COPS-like protocol.

## 8. Conclusions

The contribution of this paper is the following: (a) An intelligent traffic conditioner to mitigate the impact of RTT on the achieved bandwidth for traffic aggregates with equal target rates (b) Possible approaches to address the fairness issues between TCP and UDP traffic aggregates (c) Two intelligent traffic conditioners to distribute the excess bandwidth in over-provisioned network in proportion to the target rates.

The limitation of the above approaches are: (a) All the solutions assume one-to-one and one-to-few network topology (not one-to-any) (b) RTT-Aware and Target-Aware intelligent TCs are tied to TSW tagging algorithm. However, this can be extended to other tagging approaches as well (c) Edge nodes have to communicate among themselves to obtain certain state information.

## References

1. Floyd, S., and Jacobson, V., "Random Early Detection gateways for Congestion Avoidance", *IEEE/ACM Transactions on Networking*, V.1 N.4, August 1993, p. 397-413.
2. Blake, S. Et al, "An Architecture for Differentiated Services", RFC 2475, December 1998
3. Clark D. and Fang W., "Explicit Allocation of Best Effort Packet Delivery Service", *IEEE/ACM Transactions on Networking*, V.6 N. 4, August, 1998
4. Ibanez J, Nichols K., "Preliminary Simulation Evaluation of an Assured Service", Internet Draft, draft-ibanez-diffserv-assured-eval-00.txt, August 1998
5. Heinanen J., Baker F., Weiss W., and Wroclawski J., "Assured Forwarding PHB Group", RFC 2597, June 1999.
6. Jacobson V, Nichols K, Poduri K, "An Expedited Forwarding PHB", RFC 2598, June 1999.
7. Lin W, Zheng R and Hou J, "How to Make Assured Services More Assured", *In Proceedings of ICNP*, Toronto, Canada, October 1999



8. Yeom, I and Reddy N, "Realizing throughput guarantees in a differentiated services network", *In Proceedings of ICMCS*, Florence, Italy, June 1999
9. Mathis M, Semske J, Mahdavi J, Ott J, "The macroscopic behaviour of the TCP congestion avoidance algorithm.", *Computer Communication Review*, 27(3), July 1997
10. Elloumi O, De Cnodder S and Pauwels K, "Usefulness of three drop precedences in Assured Forwarding Service", <draft-elloumi-diffserv-threevstwo-00.txt>, Internet Draft, July 1999
11. Seddigh, N., Nandy, B., Piedad, P, "Bandwidth Assurance Issues for TCP flows in a Differentiated Services Network", *In Proceedings of Globecom'99*, Rio De Janeiro, December 1999.
12. Yeom, I and Reddy N, "Impact of marking strategy on aggregated flows in a diff-serv network," *In Proceedings of IWQoS'99*, London
13. <http://www7.nortel.com:8080/CTL/>
14. Kim H, "A Fair Marker", <draft-kim-fairmarker-diffserv-00.txt>, Internet draft, April 1999
15. Network simulator (ns-2) University of California at Berkeley, CA, 1997. Available via <http://www-nrg.ee.lbl.gov/ns/>.

# DiffServ in the Web: Different Approaches for Enabling Better Services in the World Wide Web

Hartmut Ritter, Thorsten Pastoors, and Klaus Wehrle

Institute of Telematics, University of Karlsruhe (TH),  
D-76128 Karlsruhe, Germany

{[ritter](mailto:ritter@telematik.informatik.uni-karlsruhe.de), [pastoors](mailto:pastoors@telematik.informatik.uni-karlsruhe.de), [wehrle](mailto:wehrle@telematik.informatik.uni-karlsruhe.de)}@telematik.informatik.uni-karlsruhe.de

**Abstract.** The paper discusses several approaches to use the Differentiated Services Architecture for providing better services to the users of the World Wide Web. In the future Internet a user wants to have the possibility to select between different service levels to get specific web-contents. The sender orientation of the Differentiated Services Architecture imposes some difficulties in providing the user with a means for determining the service he wants to get out of a web-server. In this paper three approaches for providing better services to the users of the World Wide Web based on the Differentiated Services Architecture are presented. Additionally, an overview over implementation issues using Linux is presented.

## 1 Introduction

Along with the enormous growth of the Internet the demands for network services have become increasingly higher. Among other things, services are required that assure certain quality of service parameters, e.g. guaranteed bandwidth, maximum packet delay or minimum variations in the inter-arrival time of consecutive packets (jitter). The World Wide Web, which can be seen as the most important and most used service in the Internet, offers only best effort service. Downloading huge files sometimes takes endlessly, because of too many packet losses resulting in long delays. A service better than best effort is needed. Providers of web services also would like to offer alternative levels of service to their customers in order to charge for higher quality. The actually most discussed approach to provide alternative service levels in the Internet is the Differentiated Services Architecture ([1]). Section 2 gives a brief introduction into the Differentiated Services Architecture.

Providing better services to the World Wide Web would mean, that the web-server responds to an HTTP request using a service better than best effort for some or all of the requests. This can be done statically, i.e. the web-server always sends using better service or distinguishes according to the source of the HTTP request or according to the requested content. This might be interesting for ISPs

which could give, for example, their clients a better access to a search engine. Nevertheless, in the common case, using the World Wide Web means to move frequently from one server to another. A normal web user does not want to use only a few good servers, but sometimes wants the possibility to get a better service of the web-server he actually visits.

The focus of this paper therefore is, how to give the user the possibility to express his actual request for a better service. The Differentiated Services Architecture, as being sender oriented, imposes some difficulties here. In a client-server scenario, typical in the World Wide Web, the request issued by the client (the web-browser) is rather unimportant with respect to quality features. The overall quality of service is determined in the first place by the service level of the response. Thus, there should be a way for the user on the client side to influence the service level of the response.

In section 3 three different approaches to overcome the problem of sender orientation are discussed. Section 4 describes the implementation under the Linux operating system. Section 5 finally presents a summary and discusses further issues. Especially the problem of pricing has to be discussed when deploying service differentiation in the Internet of the future.

## 2 Differentiated Services

The mechanisms of the Differentiated Services Architecture (DS Architecture) are held consciously simple in order to guarantee the scalability within network nodes. The architecture distinguishes three different router models.

At the boundary of the first domain acts the so-called ingress boundary node, normally the first router of the data path. Its task is the distinction of all individual traffic streams and their inspection with regard to the traffic conditioning agreement (TCA). If the TCA is exceeded, the boundary node has to form the traffic corresponding to the reservation in the TCA. In order to protect the subsequent intermediate systems from further fine granular classification the ingress boundary node records the classified service in the IP packet header by setting a codepoint in the Differentiated Services field (DS field, the former TOS field).

Normally, the ingress boundary node is the only instance that can select the service class by setting a codepoint. But the Differentiated Services Architecture [1] takes also into account that the sender can already choose the codepoint for each packet. This method is called *pre-marking*. The subsequent ingress boundary node will additionally police the marked packets in reference to the TCA. If a packet conforms to the TCA it can pass the router with the pre-marked codepoint, otherwise the packet would suffer a degradation of the service (e.g. marking as Best-Effort, discarding, etc.). Pre-Marking allows the sender – a web-server for instance – to choose the service for a flow. Therefore, some kind of reservation has to be established previously. Section 2.1 will take a closer look at reservations and management issues.

The routers of the further communication path only consider the entire aggregate of all flows of a service by looking at the DS field in the IP packet header.

In the interior of a domain, all packets are only influenced by simple queuing mechanisms, e.g. several queues - traffic shaping and policing are not carried out.

## 2.1 Management of Differentiated Services

As mentioned in [2], at least for Premium Service admission control and resource reservation are required. Furthermore, installation and updating of traffic profiles in boundary nodes are necessary. Most network administrators will not accomplish this task manually, even for long term service level agreements. Furthermore, offering *services on demand* requires some kind of bandwidth request messages and automatic admission control procedures. Therefore, the concept of *Bandwidth Brokers* was already suggested by Van Jacobson at a very early stage of Differentiated Services research [3]. In this concept, the *Bandwidth Broker (BB)* is a dedicated node in each DS domain, keeping track of the amount of available and reserved bandwidth for services, and processing admission control requests from customers or Bandwidth Brokers of adjacent domains. Additionally, it also installs or alters traffic profiles in boundary nodes.

In this paper it is assumed that a Bandwidth Broker allows to negotiate parameters like the sending rate for aggregated flows between the web-server and the web-client. The Bandwidth Broker or the corresponding ISP may also perform accounting in order to enable usage-based pricing.

## 3 Approaches for Enabling Differentiated Services in the WWW

With the way Differentiated Services are defined today, network traffic from a sender to a receiver can only be pre-marked and shaped on the sender's initiative. But often, this is not what is wanted. As stated earlier, in a typical World Wide Web scenario the user issues a short, rather unimportant (with respect to quality of service) request and awaits an important reply. Traffic from the web-server to the client - the reply - can only be pre-marked and shaped on the server's initiative. The problem that arises is obvious: How does the server know what reply is important for the client, and therefore needs a certain quality of service?

As this question deals with the client's desires, the best solution would be to let the client decide which service to use for the reply. It can most of the time better decide which quality, and hence service, would be appropriate. The main problem is therefore, how to pass the client's desires to the server. The solution should be as simple as possible, in order to avoid basic changes in the Differentiated Services Architecture, as well as in the existing client-server applications. The deployment of a dedicated signaling protocol creating overhead is also considered not to be appropriate.

Setting the codepoint used in the servers response is sufficient for changing the service level. But additionally, considering Premium Service, e.g., the specification of a rate is useful. If the aggregated rate of Premium Service traffic in

the first boundary node grows above a specified limit, packets may be delayed or discarded. Therefore, the web-server should also use an implementation of Differentiated Services that performs traffic shaping and traffic control in the same manner as in the first boundary node. In this paper, it is assumed that some kind of Bandwidth Broker provides the web-server with the information about the specified limits, e.g. the maximum rate of Premium Service flows.

In [2] some measurements have been presented, that show that the TCP protocol has extensive problems using the total amount of a reserved bandwidth. In some of the examples presented in [2] the average rate of a TCP connection over the entire transmission time was only about 20% of the reserved bandwidth. This problem can be solved in the sender with a strict traffic shaping of an outgoing TCP stream. The shaper takes care that the reserved traffic rate is not exceeded and thus, no packets will be discarded in the Differentiated Services routers.

Nevertheless, the user on the client side might not wish the server to send with the maximum rate but with a rate adapted to the local conditions. A client connected via a small-bandwidth dial-up link is constrained to the link bandwidth and does not take any advantage from a faster web-server; depending on the reservation scheme a huge percentage of Premium Service packets will be discarded at the egress node in order to avoid starvation of best-effort traffic. Considering Assured Service, the situation is similar. Thus, a second question is to be answered: How does the server know what quality of service parameters the client needs?

The following describes three different approaches that try to solve these problems under the given restrictions. For each approach its advantages as well as its disadvantages are discussed. Section 4 describes an implementation for two of the presented approaches.

### 3.1 Service Differentiation via Separate HTML Links

In the first approach, as shown in figure 1, the provider of the web-content provides several links for a specific web-content. Each link allows for downloading this web-content with a different performance when following the link. For example, a company offers five links for downloading a fifty megabyte update of its software. Figure 2 shows an example as the client would see it in his web-browser. The first link offers downloading with best effort and the next four links offer downloading with better services – whether with Premium Service or with Assured Service.

The advantage of this approach is that the user only has to distinguish between the different possibilities given in the web-page. There is no need for modifying the browser or integrating a plug-in. Each of these links (except for the first one) carries several parameters describing the service, as e.g. service level, rate, burst size, fee, etc. As soon as the customer chooses one of the links with a better service, an HTTP *Get* command is sent to the web-server – including the quality of service parameters for downloading the data. Encoding the rate and burst-size in the link of a web page would then look like the following:

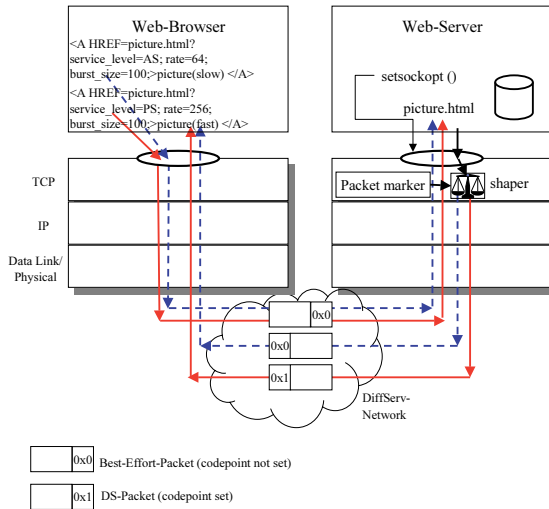


Fig. 1. Service Differentiation via Separate HTML Links

`http://www.foo.com/index.html?service=PS;rate=64;burst_size=100;`  
 This method of passing additional parameters is conform to the way an HTTP URL scheme is defined in [4].

**Downloading area for the Service Packs**  
 (supports our new DiffServ downloading feature):

- [SP1 \(BE, very slow, free\)](#)
- [SP1 \(PS, 100 kbps, 30 sec., \\$1\)](#)
- [SP1 \(PS, 500 kbps, 6 sec., \\$3\)](#)
- [SP1 \(AS, 100 kbps, ~30 sec., \\$0.75\)](#)
- [SP1 \(AS, 500 kbps, ~6 sec., \\$2\)](#)

(Service: Premium (PS) or Assured (AS) / reserved rate / estimated download time / toll billed to your credit card)

Fig. 2. Example of a Web Site offering Service Differentiation via Separate HTML Links

The web-server parses the get requests and extracts these parameters. Next, it contacts the Bandwidth Broker in order to negotiate the admitted rate and sends the data to the client. Finally, the web-server issues a system call to the TCP socket in order to set the codepoint of the outgoing packets of this connection (pre-marking). Additionally, it passes the quality of service parameters to the network subsystem. Using these parameters the network subsystem performs traffic shaping for all flows better than best-effort.

Because the service selection is coded in the HTTP request and these parameters must be passed down to the network subsystem, this approach requires changes both to the web-server application and the network subsystem of the

web-server. This can be seen as a disadvantage; on the other hand this enables the client system to remain totally unchanged. The modifications of the **Apache** web-server turned out to be minor patches (cf. section 4). Another disadvantage, however, might be that the web-pages need modifications. Multiplying all links on a web page would lead to a bad usability. Thus, this approach fits best for download areas as in the given example.

3.2 Service Differentiation via HTTP Tags

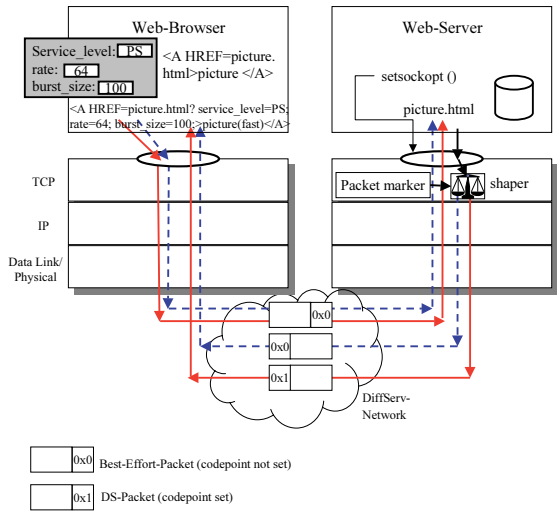


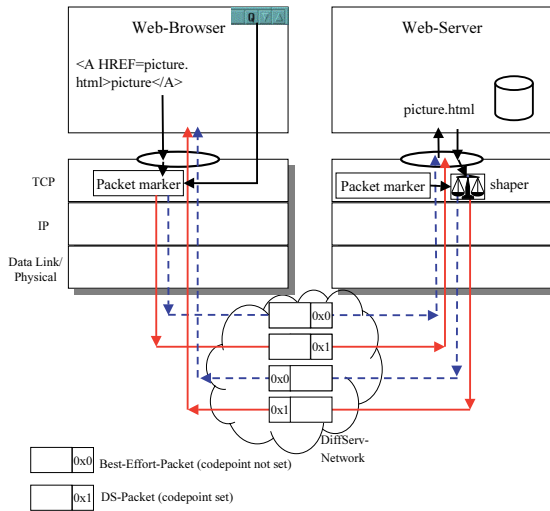
Fig. 3. Service Differentiation via HTML Tags

The second approach, shown in figure 3, is similar to the previous one. However, the way the user chooses the quality of a specific web-content differs. Using this approach, the web-page only provides a single link for a specific web-content as it usually does today. With a click on the link with the left mouse button, the content is transferred as it is today - with best effort. On a click with the right button a menu pops up, with an additional entry for getting this web-content with some better service. Choosing this entry pops up another window in which the user specifies the rate and burst-length with which he wants to receive the content. From there on, this approach behaves like the previous one. An **HTTP Get** command is sent to the web-server - including the parameters specified by the user. The web-server negotiates the parameters with the Bandwidth Broker and sends the data to the client as described above.

The main difference in this method compared to the previous one is the fact, that here, the web-client needs to be modified: The additional field in the pop-up window, as well as the window for submitting rate and burstlength have to

be integrated. The web-pages however, could be the ones we have today. Thus, doubling the links can be avoided. This is obviously an advantage. Furthermore, this approach is also more flexible with respect to choosing the quality of service parameters. On the other hand, this might be a very demanding task for normal users, as a deeper understanding of the parameters would be necessary. A user who is not informed about its actual link bandwidth will have problems to choose the appropriate values.

### 3.3 Service Differentiation via Packet Marking



**Fig. 4.** Service Differentiation via Packet Marking

In this approach the server system sets the codepoint in the DS field to the appropriate value according to the codepoint of the IP-packets of the client request. The codepoint of the client request is set by the user of the web-browser. As stated before, this pre-marking of packets is described in the Differentiated Services Architecture. Due to the small bandwidth required by an HTTP request, the request is very likely to be passed across the network to the web-browser. The server system, where the web-server is running on, detects incoming packets with the codepoint set. This codepoint is then copied to all packets sent by the server on this TCP connection. By this simple means, the client system can determine the service level of the response of the web-server.

To express his current preferences, the user can not use different HTML-links on the web site. Instead, the user needs another interface for expressing his or her current preferences. One approach for a simple interface, the **Q-Button**, is presented in [5]. In the corner of the web-browser window an additional button is integrated: This button is labeled with the letter **Q**, denoting Quality of



Service. It is not part of the application, but of the window manager. The user expresses the current preference for an application running in the related window by clicking on the Q-Button in the window bar. If the user presses the button, this triggers a packet marker in the kernel to mark outgoing packets of the TCP connections of this application.

The advantage of this approach is its simplicity: The simple interaction of pressing a button results in a better service. A service degradation to best effort can be achieved by right-clicking the button. Changes to the web-pages or to the web-browser as in the other two approaches are not necessary. The web-server application also remains unchanged, as the packet marking is done in the network subsystem of the server. Thus, the approach requires no changes to the applications, but the integration of a packet marker into the network subsystem, which can be done quite easily, at least under Linux. The approach fits best for dynamic changes of the service level. Requesting a better service neither requires another parameter menu nor must the web-page support this explicitly.

The answer to the second question (“How does the server know what quality of service parameter the client needs?”) however is more problematic. The concept does not allow passing a set of technical parameters like the desired rate to the server. Explicit signaling for every short-term connection is very likely to create too much overhead. Depending on the future development of the Differentiated Services Architecture some feedback to the web-server might be possible, be it by simple backward congestion indication or be it by a notice from the Bandwidth Broker. In the mean time, this approach seems to fit best for short interactions, typical in many World Wide Web sessions. The download of huge files, where more detailed quality of service parameters are needed, is not supported with fine granularity.

### 3.4 Comparison of the Approaches

The presented approaches differ in the areas of usability, granularity of service parameter selection and in terms of changes necessary to the network subsystem of the operating system or to the applications. A summary of the differences is given in figure 5. Especially the possibility to determine the quality of service parameters of the web-server makes a difference. As indicated before, the specification of a parameter set will be useful when downloading huge files or requesting a multimedia stream. On the other hand, this specification is demanding for the user and creates too much overhead in short interactions. Up to now, it is not clear what exact kind of mechanisms supporting short-term reservations will take place in the Differentiated Services Architecture; thus, the tradeoff between usability and more detailed parameter specification can not be decided yet.

## 4 The Implementation

A prototypic implementation using the Linux operating system was done. The current implementation is based on the kernel version 2.2.5. Also changes to the

	Service Differentiation via separate HTML Links	Service Differentiation via HTTP Tags	Service Differentiation via Packet Marking
Usability	easy to use	Demanding for the user	easy to use
Granularity of service selection	coarse	fine	very coarse
Changes to applications	web-server	web-server and web- browser	any
Changes to operating system	on server side	on server side	on server and client side

**Fig. 5.** Comparison of the Presented Approaches

web-server had to be made. As for Linux, the Apache web-server is available in source code and runs successfully on thousands of computers. The version number of the Apache web-server was 1.3.9.

The implementation was tested in the UNIQuE testbed described in [6]. The UNIQuE testbed comprises a fully equipped Differentiated Services Domain and several end systems. The realization of Differentiated Services support in the software routers of the testbed is based on the KIDS implementation (cf. [7]).

In the remainder of this section, the elements of the implementation are described. A small set of building blocks is sufficient to realize all described approaches. The modifications of the web-browser, as necessary for the second approach described in section 3.2, have not yet been implemented. The feasibility of this approach seems clear when realizing the first approach given in section 3.1.

#### 4.1 Packet Marking

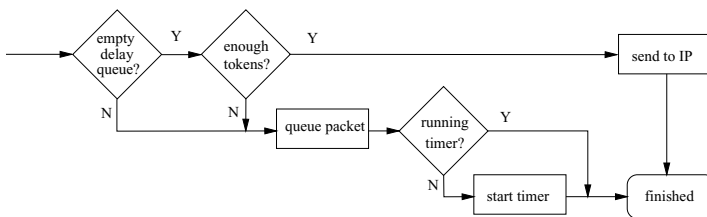
As indicated by its name, the Packet Marker marks outgoing traffic; i.e. sets the codepoint in the DS field of the IP packet. The input of the packet marker differs between the approaches: Whereas in the first and second approach the desired codepoint is given to the packet marker from the web-server application, in the third approach the packet marker gets this input from the incoming packets. In both cases, already existing structures in the Linux kernel could be used: For each HTTP `Get` request a new TCP connection is opened and a socket is created. Associated with each socket is a variable called `ip_tos` that controls the DS field of outgoing IP packets. Packet marking is done via changing the value of `ip_tos` to the desired codepoint; all IP packets leaving this socket now will get the new `ip_tos` value in the DS field of the header. Therefore, the modification needed was either to copy the codepoint of the incoming IP packet to this variable or to allow the web-server application to set it. The latter can be done via the system call `setsockopt`.

## 4.2 DS Profile

As mentioned before, traffic shaping can already be done in the host. Doing this, allows for fine granular classification and avoids packet discards at the boundary node. When performing shaping of a traffic flow (TCP-traffic in the case of the web-server), the appropriate place is in the transport layer. The implementation is based on **shaping sockets**, i.e. the shaping of the traffic flows is associated with a socket and done inside the TCP layer of the TCP/IP protocol stack. Therefore, the socket structure was extended with a pointer to a DS profile used by the traffic shaper. The DS profile contains the parameters **rate** and **burst\_size** which can be set from user-space by an extension of the **setsockopt** system call. In order to avoid collisions with the current TCA, these values can be checked with the Bandwidth Broker and can be downgraded if necessary.

## 4.3 Traffic Shaping

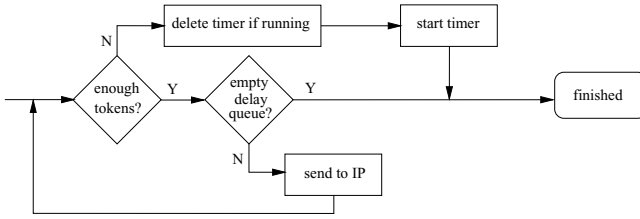
To shape the traffic according to the desired rate and burst-size, a token bucket is used. The parameters **rate** and **burst\_size** stored in the DS profile are therefore transformed to token rate and token bucket size. Besides these parameters, the DS profile keeps also internal parameters like the variable **last\_sent** which determines the last send time and a pointer to the delay-queue. The token bucket implemented is a byte-oriented token bucket to achieve higher accuracy. As it works with TCP-packets, a packet-oriented bucket would not be precise enough as their length can vary extremely. Two functions are required for shaping. The first one (**ps\_shaping()**) is called before we pass the packet from TCP to IP (cf. figure 6). It checks for tokens and either sends the packet immediately or queues it. The second one (**delayed\_xmit()**) processes the delay-queue and is invoked on demand by a timer (cf. figure 7).



**Fig. 6.** The Function **ps\_shaping()**

The token bucket algorithm is realized as follows. In **ps\_shaping()** the packet arrival time **current\_stamp** is taken. The amount of new tokens that were theoretically filled into the bucket between **last\_sent** and **current\_stamp** is calculated by

$$\text{new\_tokens} := (\text{current\_stamp} - \text{last\_sent}) / \text{cycles\_per\_byte},$$



**Fig. 7.** The Function `delayed_xmit`

with

$$\text{cycles\_per\_byte} := 8 * \text{cpu\_clock} / \text{rate}.$$

If now, there are enough tokens to send the packet immediately and the delay-queue is also empty, the amount of tokens in the bucket is updated by adding the new tokens, pruning the tokens against the bucket-size and subtracting the packet-length. `last_sent` is set to `current_stamp`. If there are not enough tokens available or the delay-queue is not empty, the packet is enqueued at the end of the delay-queue and a timer for `delayed_xmit()` is started. Unfortunately, timer in the Linux kernel can only be started with a accuracy of jiffies. Thus, one can not exactly calculate the expiration time  $t_{exp}$  which is calculated by:

$$t_{exp} := (\text{packet-length} - (\text{tokens} + \text{new\_tokens})) / \text{bytes\_per\_jiffie},$$

$$\text{bytes\_per\_jiffie} := \text{rate} / (8 * 100)$$

However, a timer is only started if no one is already running. The function `delayed_xmit()` does almost the same as `ps_shaping`, except that it processes the delay-queue starting at the head and sends packets as long as there are enough tokens available. If it empties the queue, it stops. If not enough tokens are available, a new timer is started, with the time of expiration calculated as above. Notice, that a running timer is stopped before, as it might not have been calculated on base of the packet at the head of the delay-queue.

## 5 Summary

In this paper three different approaches to enable better services in the World Wide Web were presented. Enabling better services faces the basic problem of the sender orientation of the Differentiated Services Architecture. This conflicts with the typical structure of the client-server interaction in the World Wide Web. The overall quality of service brought to the user at the client side is determined mainly by the quality of service of the response. Therefore the user on the client side must be provided by a means to influence the service level of the server's response. Three approaches to overcome this problem were presented; designed under the constraints of the given Differentiated Services Architecture. These approaches differ in the complexity and location of changes. Also the ease of use for the user differs. As shown, there is a tradeoff between ease of use and an exact determination of parameters by the user.

## 5.1 Future Topics

The provision of better services in the World Wide Web using the Differentiated Services depends on two factors: The future management architecture of services from end to end is not yet clear, and connected with this, the same can be said about the problem of reservations. The current development of Differentiated Services focuses on more static reservations, but from the beginning on [3], as services would evolve, more dynamic service level agreements were envisioned. Depending on this evolution, the presented approaches might need further adaptation and modification. The sometimes discussed deployment of reservation protocols like RSVP with DiffServ-Networks e.g. would have a strong impact on the provision of end-to-end services in the context of the Differentiated Services Architecture.

## References

1. Baker, F., Heinanen, J., Carlson, M., Davies, E., Wang, Z., Weiss, W.: An Architecture for Differentiated Services. RFC 2475, IETF (1997)
2. Bless, R., Wehrle, K.: Evaluation of Differentiated Services using an implementation under Linux. In: Proceedings of the *7th IFIP Workshop on Quality of Service*, IEEE (1999)
3. Nichols, K., Van Jacobson, Zhang, L.: A Two-bit Differentiated Services Architecture for the Internet. RFC2638, IETF (1999)
4. Berners-Lee, T., Masinter, L., McCahill, M.: Uniform Resource Locators (URL). RFC1738, IETF (1994)
5. Bechler, M., Ritter, H., Schiller, J.H.: Quality of Service in Mobile and Wireless Networks: The Need for Proactive and Adaptive Applications. In: Proceedings of the 33rd Annual IEEE Hawai'i International Conference on System Sciences (HICSS), IEEE (2000)
6. Bechler, M., Ritter, H., Schiller, J.H.: Research in the UNIQuE Environment: Mobile Multimedia Services. In: Proceedings of the 3rd IASTED/ISMM Conference on Internet and Multimedia Systems and Applications (IMSA'99), IEEE (1999)
7. Karlsruhe Implementation of Differentiated Services (KIDS) homepage, <http://www.telematik.informatik.uni-karlsruhe.de/Forschung/KIDS.html> (1999)

# Performance of Short TCP Transfers

Chadi Barakat and Eitan Altman

INRIA, 2004 route des Lucioles, 06902 Sophia-Antipolis Cedex, France  
{cbarakat,altman}@sophia.inria.fr

**Abstract.** Many works have studied the negative impact of slow start on the performance of short transfers. Some works propose to accelerate the window increase during this phase in order to improve the performance especially on satellite links. Others propose to set the slow start threshold at the beginning of the connection to a more accurate value in order to avoid losses. But, these works didn't account for the impact of network buffers on the performance. It is known that small buffers along with a fast window increase leads to an early buffer overflow and an underestimation of the network capacity. This may change completely the performance predicted by these modifications. In this paper, we present a general analysis of this first phase as a function of all the possible parameters. We show that, as claimed, the previous works improve the performance on paths with large buffers. However, on paths with small buffers, completely different results could be obtained.

## 1 Introduction

TCP is the main responsible for the stability of the Internet. By varying the size of its window ( $W$ ), it controls the flow of application packets so as to avoid network congestion [10]. The two main algorithms used by TCP for congestion control are Slow Start (SS) and Congestion Avoidance (CA) [10,13]. SS is used to increase quickly  $W$  until a certain estimate of the network capacity. By network capacity or pipe size, we mean in the sequel the maximum number of packets that can be fit on the path. The network capacity estimate is called the SS threshold ( $W_{th}$ ). Once  $W_{th}$  is reached, the source switches to CA where  $W$  is increased slowly to probe the network for any extra bandwidth. At any moment, the window increase is halted once the network gets congested. Congestion is detected via losses. Here, TCP sets  $W_{th}$  to half the current window, reduces its window, recovers from losses and starts again its window increase.

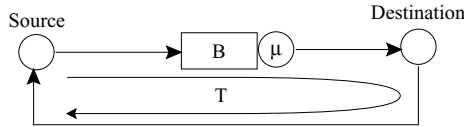
With SS, TCP is supposed to fill quickly the network capacity. If  $W_{th}$  is correctly set, the source switches to CA before the congestion of the network. If  $W_{th}$  overestimates the network capacity, congestion occurs during SS and  $W_{th}$  is set to a more accurate value. SS functions in this case as a quick means to gauge the network capacity. Particularly, we see this behavior of SS at the beginning of the connection where  $W_{th}$  is set to a default value usually equal to the receiver capacity (i.e. the window advertised by the receiver). But, this measurement of the network capacity is not without cost. Due to the fast window increase, SS

overloads the network and causes many losses. A long Timeout and a new SS are required to recover from these losses [7]. For this reason, a proposition to set  $W_{th}$  at the beginning of the connection to a more accurate has been made in [9]. The author proposes to set  $W_{th}$  to the Bandwidth-Delay Product (BDP) of the connection path.

TCP uses the flow of Acknowledgments (ACK) as a clock to increase  $W$  and to trigger the transmission of packets.  $W$  is increased by one packet for every ACK during SS and by one packet for every window's worth of ACKs during CA. When an ACK is received, the source transmits in a burst as many packets as its window allows. This results in a source transmission rate higher than the available bandwidth in the network when the window is increased quickly as during SS. If network buffers are not well dimensioned, losses will appear early before filling the pipe between the source and the destination or even before reaching a correctly-set SS threshold [4,6,11]. This results in a wrong congestion signal and an underestimation of the network capacity. This is an important problem in networks with small buffers compared to their BDP. A typical example of such networks are satellite networks where the BDP is important and where there are many limitations on the buffer size on satellite board.

The impact of buffer size on SS has been studied in many works [4,6,11]. These works consider a long TCP-Tahoe connection [7,10] where SS is called after every loss detection and where  $W_{th}$  is a correct estimate of the network capacity. The aim of SS in this case is to reach  $W_{th}$  quickly and without losses. They don't consider the case where it is to SS to gauge the network capacity. They describe the problem and find an expression for the minimum buffer size required to absorb SS burstiness. However, the recent versions of TCP (Reno, SACK) [7] try to avoid SS during the steady state of the connection. The connection is supposed to stay in CA where the problem of early buffer overflow doesn't exist. We believe that the problem still exists for short transfers since TCP is obliged to start any transfer with a SS phase. Moreover, short transfers dominate most of today Internet traffic mainly due to HTTP traffic. Such transfers are in general of interactive type (e.g. Web transactions) and they are very sensitive to the service provided by the network and the underlying protocols.

In this paper, we study the impact of network buffers on the first SS phase of a TCP connection. We consider also the impact of the aggressiveness of TCP during SS. By aggressiveness or burstiness of SS, we refer in the sequel to how fast the window is increased during SS. We try to answer two main questions. First, given a certain bandwidth, a round-trip time (RTT) and a buffer size, how to set  $W_{th}$  in order to avoid losses. We show that the value to give to  $W_{th}$  is a function of all network parameters not only the BDP as suggested in [9]. Second, we consider the case where it is to SS to gauge the network capacity. We analyze in this case the effect of network parameters and the window increase rate on the capacity estimate provided by SS. We show that, for small buffers, this estimate decreases when we increase the aggressiveness of SS leading to an overall performance deterioration. This tells us that solutions like Byte Counting [1] that accelerate the window increase during SS in order to improve the performance



**Fig. 1.** The network model

may deteriorate the performance instead of improving it if network buffers are not enough large. We present guidelines for how to increase  $W$  during SS. Based on this analysis, we propose a new window increase algorithm that reduces the duration of SS while not overloading network buffers.

In the next section, we outline our analytical model for the evaluation of TCP performance. In section 3, we study the impact of the value given to  $W_{th}$ . Section 4 studies the case where  $W_{th}$  is set to a high value and where it is to SS to estimate the network capacity. In section 5, we extend our analysis to the case of multiple connections sharing the same path. Throughout the paper, analytical results are validated with a set of simulations using `ns`, the Network Simulator developed at LBNL [12]. The work is concluded in section 6.

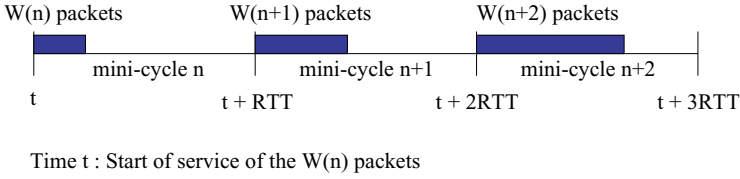
## 2 A Model for TCP Performance

Consider a TCP connection that transfers files of size  $S$ . The widely used Reno version of TCP [7,13] is used throughout the paper. However, the analysis can be applied to the other versions as well. We model the network with a single node of rate  $\mu$  packets/s and of Drop Tail buffer of size  $B$  packets (Figure 1).  $T$  denotes the constant component of the RTT. This model has been often used in the literature to study the performance of TCP [1,4,6,8,11].

The performance of a short TCP transfer is strongly dependent on TCP behavior during SS. The key point in the characterization of such behavior is the calculation of the window at which losses occur during SS assuming  $W_{th}$  is set to an infinite value. We call this window *the overflow window* and we denote it by  $W_B$ . It gives us an upper bound on  $W_{th}$  since the source must get in CA before reaching  $W_B$  if it wants to avoid losses during SS. In case of losses during SS, it determines the window evolution after the recovery from losses. The new estimate of the network capacity which we denote by  $W'_{th}$  and which is equal to the SS threshold after the recovery from losses is a direct function of  $W_B$ .

Normally,  $W_B$  is equal to the pipe size ( $B + \mu T$ ) which has been assumed implicitly in [1,9]. However, in [4,6,11], the authors have shown that  $W_B$  can take a small value in case of small buffers. However, neither the BDP nor the window increase rate has been considered. Here, we find the general expression of  $W_B$ . It is upper bounded by the pipe size and it is a decreasing function of the window increase rate during SS.





**Fig. 2.** Bursts at the output of the bottleneck

## 2.1 A Model for TCP Aggressiveness During Slow Start

Let  $W(t)$  denote the window size in packets at time  $t$ . We suppose that after one RTT, the window is increased during SS by  $W(t)/d$  packets.  $d$  can be the result of the receiver delaying ACKs and sending an ACK for every  $d$  packets and of the sender increasing its window by one packet for every non-duplicate ACK (Standard TCP [13]).  $d = 1$  means that the receiver is acknowledging all the packets and  $d = 2$  represents the *delay ACK* mechanism widely implemented in TCP receivers [13].  $d$  can also account for any window increase policy at the source different than that of Standard TCP (STCP).

## 2.2 The Overflow Window $W_B$

As in [4,11], we divide SS into mini-cycles (MC). The duration of a MC is equal to the current RTT. Let  $W(n)$  be the number of packets transmitted during MC  $n$ . The next MC starts when the ACK for the first packet of these  $W(n)$  packets reaches the source. The window size during the next MC is equal to,

$$W(n+1) = W(n) + W(n)/d = \alpha W(n), \quad (1)$$

with  $\alpha = (d+1)/d$ .

We consider that STCP with non-delayed ACKs is the most aggressive case ( $d \geq 1$ ). Suppose that the recurrent relation (1) is valid for every  $n \geq 0$ . Suppose also that SS starts with a window equal to one packet. Thus,

$$W(n) = \alpha^n W(0) = \alpha^n.$$

Packets are transmitted in long bursts at a rate higher than  $\mu$ . They wait in  $B$  until they are served. A burst of length  $W(n)$  is served during MC  $n$  and it is followed by an idle period until the arrival of the burst of the following MC (Figure 2). This idle time between bursts disappears when  $W$  exceeds the BDP ( $\mu T$ ). Given that the number of packets transmitted during a MC increases by a factor  $\alpha$ , we can suppose that the long bursts transmitted by the source have an average rate  $\alpha\mu$ .

When a long source burst reaches the bottleneck, a queue starts to build up in  $B$  at a rate  $\alpha\mu - \mu = \mu/d$ . Two cases here must be considered. The first case is when  $B$  doesn't contain any packet from the previous MC when the first packet of the burst of the current MC reaches the bottleneck. The second case is when

some packets from the previous MC are still waiting in  $B$ . In [4,6,11], the first case has been only considered.

In the first case, a burst of size  $B(d+1)$  is required to fill the buffer. Let  $n_B^1$  be the number of the MC during which  $B$  overflows. The number of packets transmitted during this MC must be larger than  $B(d+1)$ . But, the number of packets transmitted during the previous MC must be less than  $B(d+1)$  otherwise the overflow would have occurred during the previous MC. Thus,  $n_B^1$  satisfies,

$$\alpha^{n_B^1-1} < B(d+1) \leq \alpha^{n_B^1}.$$

From the first case, we have also,  $\alpha^{n_B^1-1} < \mu T$ . According to our definition of  $d$ , the transmission of a burst of  $B(d+1)$  packets requires an increase in  $W$  by  $B$  packets since the beginning of MC  $n_B^1$ . It follows that,

$$W_B = W(n_B^1 - 1) + B = \alpha^{n_B^1-1} + B. \quad (2)$$

Now, we consider the second case. The window size is larger than  $\mu T$ . The burst size required to fill the buffer is less than  $B(d+1)$  since there are some packets waiting from the previous MC. It is simply equal to the number of empty places at the beginning of the MC times  $(d+1)$ . The increase in the window between the beginning of the MC and the overflow is equal to the number of empty places. Suppose that the overflow happens during MC  $n_B^2$ . Then,  $W_B$  changes and becomes equal to,

$$W_B = W(n_B^2 - 1) + B - (W(n_B^2 - 1) - \mu T) = B + \mu T. \quad (3)$$

Two expressions for  $W_B$  are then available. If the window size during MC  $n_B^1 - 1$  is less than  $\mu T$ , then  $W_B$  will be given by equation (2), otherwise it will be given by equation (3). We can combine these two expressions into a single one as mentioned in the following theorem.

**Theorem 1.** *If Slow Start is not terminated before the occurrence of losses, the buffer at the entry of the bottleneck link will overflow at a window,*

$$W_B = B + \min(\mu T, \alpha^{n_B-1}),$$

with  $n_B$  given by  $\alpha^{n_B-1} < B(d+1) \leq \alpha^{n_B}$ .

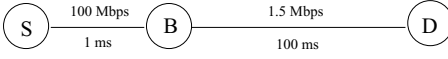
The following corollary can be directly derived.

**Corollary 1.** *The bottleneck buffer will not overflow during Slow Start if  $W_{th}$  is set to less than the  $W_B$  given by Theorem 1.*

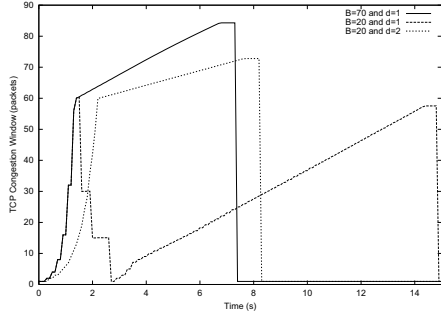
To simplify the analysis, we approximate  $\alpha^{n_B}$  by  $B(d+1)$ . The same approximation has been made in [4,11]. The expression of  $W_B$  becomes,

$$W_B = B + \min(\mu T, Bd). \quad (4)$$

It is clear that this window is equal to the pipe size  $B + \mu T$  whereas  $B$  is larger than  $\mu T/d$ . Once  $B$  becomes less than  $\mu T/d$ , SS becomes unable to fill the network capacity. This is where the problem of early buffer overflow appears.



**Fig. 3.** The simulation scenario



**Fig. 4.** TCP congestion window vs. time

### 3 Impact of $W_{th}$ on the Performance

The correct value for  $W_{th}$  (i.e. just less than  $W_B$ ) is a function of all the parameters not only  $\mu$  and  $T$ . It decreases with the decrease in the buffer size or the increase in TCP burstiness. If the buffer size is less than  $\mu T/d$ , it becomes independent of the available bandwidth! If we take as an example the value proposed for  $W_{th}$  in [9] (i.e. the BDP), we find that a  $B$  larger than  $\mu T/(d+1)$  is required for this proposition to work otherwise losses will not be avoided.

Consider the simulation scenario in Figure 3. The source transmits a file of size 100 KB. TCP packets are of size 512 Bytes. We give two values to  $B$ , 70 packets which is approximately equal to the BDP and 20 packets. For a  $W_{th}$  equal to 50 packets, we plot in Figure 4 the congestion window as function of time. Three cases are considered,  $B = 70$  packets and  $d = 1$ ,  $B = 20$  packets and  $d = 1$ ,  $B = 20$  packets and  $d = 2$ .  $d$  is implemented by simply delaying ACKs at the receiver. Normally, for such a threshold smaller than the BDP, one must predict that losses will not appear during SS. This is true for  $B = 70$  but it is not the case for a buffer of 20 packets and a  $d = 1$ . A decrease in TCP burstiness is required (from  $d = 1$  to  $d = 2$ ) to help the buffer to absorb the bursts of SS.

### 4 Case of a High Slow Start Threshold

In this section, we study the case where TCP uses SS to gauge the network capacity. The performance is a function of  $W'_{th}$ , the capacity estimate after the recovery from losses which is a direct function of the overflow window.

#### 4.1 Calculation of $W'_{th}$

The buffer overflow during SS is detected one RTT after its occurrence. During this RTT,  $W$  increases from  $W_B$  to  $\alpha W_B$  unless the source gets in CA. This later case corresponds to  $W_B < W_{th} < \alpha W_B$ . Congestion detection happens at a window  $W_D$  equal to,

$$W_D = \min(W_{th}, \alpha W_B). \quad (5)$$

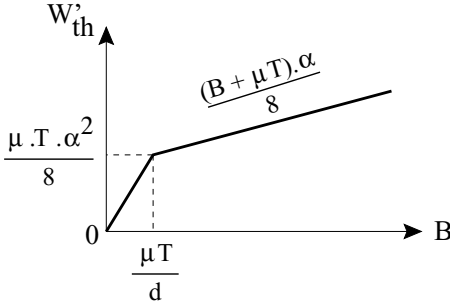
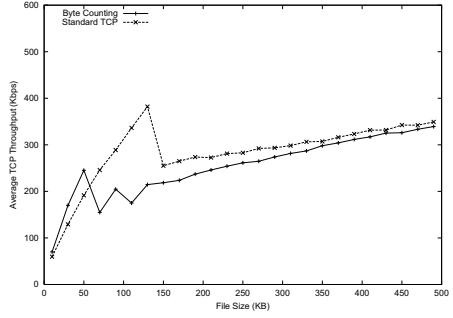


Fig. 5. New estimate vs. buffer


 Fig. 6. BC vs. STCP for  $B = 20$ 

Here, TCP sets  $W$  and  $W_{th}$  to half  $W_D$  and starts to recover from losses. Most often, it succeeds to detect the first two losses via duplicate ACKs [7]. However, the third and subsequent losses require a Timeout to be detected [7]. Since Reno divides its window by two upon every loss detection [7,13], the SS threshold after the Timeout is set to one eighth  $W_D$ . It is set to one fourth  $W_D$  when the second loss cannot be detected via duplicate ACKs. In the sequel, we assume that  $W'_{th}$  is equal to  $W_D/8$ . Also, we suppose that  $W_{th}$  is set higher than  $\alpha W_B$  so that the congestion is always detected at a window  $W_D = \alpha W_B$ .

#### 4.2 Interaction Between Buffer Size and SS Aggressiveness

We study here the impact on the performance of the window increase rate during SS. We try to find where a given rate leads to an improvement in the performance and to define the optimum window increase strategy that works under different buffer sizes. The objective is always to reduce the duration of SS. With the increase in BDP and RTT (i.e. satellite networks), the acceleration of SS is becoming one of the main requirements for a good performance [2,3,5]. We assume that the window increase rate is adjusted during SS without affecting the CA phase. The easiest way to implement such kind of strategies is to work at the source since it is the only entity in the network able to distinguish between SS and CA. An example is Byte Counting (BC) [1] proposed to overcome the negative impact of the delay ACK mechanism ( $d = 2$ ) on the performance of short transfers. Upon the receipt of an ACK, the window is increased during the first SS phase by the number of packets covered by the ACK rather than by one packet with a maximum window increase of two packets. This is equivalent to reducing  $d$  from 2 (case of STCP) to 1.

If the buffer size is large enough to absorb SS bursts, any change in  $d$  will not change the overflow window which remains equal to the pipe size. An increase in the aggressiveness in this case improves the performance since it reduces the time taken by SS without changing the estimate. This happens whenever  $B$  is larger than  $\mu T/d$ .

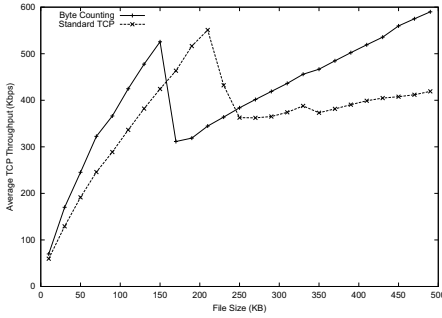
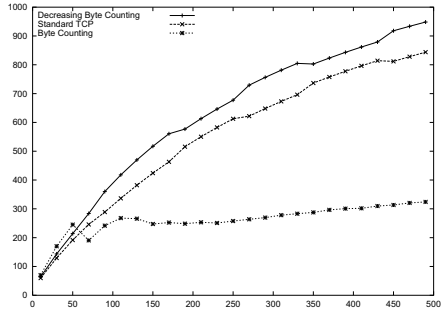
Fig. 7. BC vs. STCP for  $B = 70$ 

Fig. 8. Performance of DBC

The problem exists when the buffer is smaller than  $\mu T/d$ . As we see in Figure 5, the network capacity estimate deteriorates in this case with any increase in aggressiveness due to a deterioration of the overflow window which becomes less than the pipe size. The source gets then in CA at a small window and requires a long time to compensate the resulting capacity underestimation. It is better to stop increasing the aggressiveness of SS once  $B$  becomes less than  $\mu T/d$ . In Figures 6 and 7, we compare BC to STCP under two buffer sizes (20 and 70 packets). The throughput is plotted as a function of the file size. For a large buffer, BC works perfectly and gives better performance. However, for a small buffer, BC is so aggressive that it fills the buffer before filling the pipe. This gives lower estimate and thus lower performance than STCP for most of the file sizes although STCP adopts a slower window increase.

The problem with BC (or other similar strategies) is that it uses the same  $d$  for the entire SS phase. However, the best performance is obtained when the source starts with a small  $d$  then switches to a larger one just before the overflow of  $B$  and it continues like this until the pipe is filled or  $W_{th}$  is reached. This consists in reducing SS burstiness with the increase in  $W$ . Such mechanism is difficult to implement given that TCP is not aware of the buffer size in network nodes. In the next section, we propose a possible solution to this problem that we call *Decreasing Byte Counting (DBC)*. It has the advantage of not requiring the buffer size but rather an idea on the ratio of the buffer size on the path to the BDP.

### 4.3 Decreasing Byte Counting

Fixing the same  $d$  during SS is not the optimal solution since at the beginning, the problem of aggressiveness is not as pronounced as at the end.  $d$  must be incremented gradually during SS in order to push the congestion until the overflow of the pipe. Starting at  $d = 1$ ,  $d$  must be set to 2 when  $W$  reaches  $2B$  (equation (4)), then to 3 when  $W$  reaches  $3B$ , then to 4 when  $W$  reaches  $4B$ , etc. If we consider a continuous variation of  $d$ , our analysis shows that this factor must be increased linearly with  $W$  and inversely proportional to  $B$ . There is a certain

minimum limit on this factor which we fix in this work to 1. But the buffer size is not known at the source. All that the source knows is its SS threshold. Thus, instead of using  $B$ , we propose to use another factor which accounts for the maximum value of  $d$  that it seems to be enough to reach the chosen  $W_{th}$  on a given path. Call this value  $d_{max}$ . It is a function of the ratio of the buffer size to the BDP and of the ratio of the value given to  $W_{th}$  to the BDP. If  $W_{th}$  is set at the beginning of the connection to the BDP as proposed in [9],  $d_{max}$  will be only a function of the ratio of  $B$  to BDP. Given this  $d_{max}$ , we vary  $d$  linearly between 1 and  $d_{max}$  as long as  $W$  grows from 1 (or other initial value) to  $W_{th}$ . This gives us,

$$d(W) = 1 + (d_{max} - 1)(W - 1)/(W_{th} - 1).$$

A possible value for  $d_{max}$  can be that of STCP ( $d_{max} = 2$ ). In this case, BC overloads the network buffers whereas STCP does not. This is equivalent to applying BC at the beginning of SS then in starting to get out of BC towards STCP as long as  $W$  grows. Our solution should give better performance than STCP and BC in this region. In the region where STCP overloads the network buffers (very small buffers), a  $d_{max}$  larger than 2 is required. Taking  $d_{max} = 2$  should give poorer performance than STCP in this case but better performance than BC. Finally, one should expect that in the region where BC doesn't overload the network buffers (large buffers), BC should give the best performance.

We show in Figure 8 a comparison between BC, STCP and our proposition.  $W_{th}$  is set to the BDP. The simulation scenario is the same as that of Figure 3. A buffer size of 30 packets is taken. With such buffer, BC is unable to reach the BDP whereas STCP is. We see well how BC causes losses and reduces the performance w.r.t. STCP. Our proposition however is able to increase faster the window while avoiding losses. It provides the best performance with respect to the two others.

## 5 Case of Multiple TCP Connections

We describe here briefly the behavior of a new TCP connection that arrives to a network crossed by another TCP traffic. Also, we verify the benefit of our DBC algorithm in the context of many concurrent TCP connections.

### 5.1 A Model for the Case of Multiple Connections

It is known that in case of Drop Tail buffers and in case of close RTT, the TCP connections sharing the same bottleneck change their windows in a synchronized manner [11]. A congestion event causes losses from all connections forcing them to reduce their windows simultaneously. Suppose that all the running connections have the same RTT. Thus, the total number of packets in the network varies periodically between half the pipe size and the pipe size [11]. This behavior is independent of the number of active connections.

Suppose that a new connection arrives at a random time. Thus, it will see in the network a number of packets between half the pipe size and the pipe size. Call  $N$  the number of packets it finds. Using  $N$ , we will try to find the

parameters of the equivalent single-node network seen by the new connection. Once these parameters are calculated, we can apply our previous analysis to characterize the behavior of the first SS of the new connection.

If  $N$  is smaller than  $\mu T$ , the new connection will see an empty buffer together with  $N$  packets propagating on the link (i.e. not waiting for service in a queue). This is because the other connections are operating in CA where no queue builds up in  $B$  until  $N$  exceeds the BDP [11]. The equivalent network seen by the new connection is formed of a buffer  $B$  and a BDP equal to  $\mu T - N$ . Now, if  $N$  is larger than  $\mu T$ , the new connection will see a full link together with  $N - \mu T$  packets waiting their service in  $B$ . In this case, the equivalent network is formed only of a buffer of size  $B + \mu T - N$  packets. Thus, the overflow window given in equation (4) for the single connection case can be rewritten in the case of multiple connections as,

$$W_B = B + \min(\mu T - N, Bd). \quad (6)$$

This overflow window takes its value between zero and a maximum value we call  $W_B^{max}$  which corresponds to a number of packets in the network equal to  $N = (B + \mu T)/2$ .  $W_B^{max}$  is equal to,

$$W_B^{max} = B + \min((\mu T - B)/2, Bd). \quad (7)$$

We see well that  $W_B^{max}$  moves to zero when  $B$  moves to zero. The performance is again an increasing function of  $B$ . But, we notice that the impact of  $d$  is less important in this case than in the case of a single connection. Indeed, in the case of multiple connections, the equivalent network seen by the new connection has the same buffer size ( $B$ ) as the real network but a smaller BDP ( $\mu T - N$ ). The real BDP is shared by the different connections whereas the buffer can be considered as dedicated to the new connection.

In the case of multiple connections, an early buffer overflow occurs when  $Bd < \mu T - N$ . It never occurs if the buffer size satisfies  $Bd > \mu T - N$  when  $N$  is equal to  $(B + \mu T)/2$ . This corresponds to a buffer size larger than  $\mu T/(2d + 1)$  which is less important than the buffer size required in the case of a single connection. Now, even if the buffer size is smaller than  $\mu T/(2d + 1)$ , the problem does not always occur. It is not seen when  $N$  is close to  $\mu T$ .

To show this behavior in presence of multiple connections, we simulate the scenario in Figure 9. Five sources share a 7.5Mbps bottleneck link. Every source has many files to transmit. The file size is chosen randomly between 100KB and 1MB. Files of a source are transmitted on successive TCP connections. These connections are separated by a random time between 0 and 5 seconds. We run 50 simulations of 50 seconds each then we calculate the average TCP throughput during a transfer.

We plot in Figure 10 the throughput of a transfer as a function of  $W_{th}$ . Two cases are considered:  $B = 70$  packets and  $d = 1$ ,  $B = 20$  packets and  $d = 1$ . The receiver sends an ACK for every other packet and the sender changes its window increase only during SS. The results in this figure match well equations (6) and (7). Theoretically, for these cases,  $W_B^{max}$  is equal respectively to 106KB and

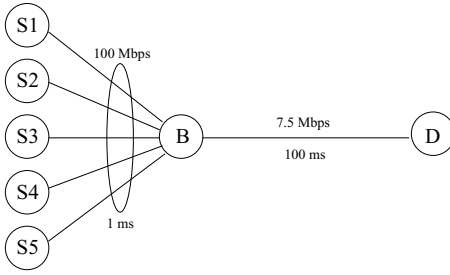


Fig. 9. The simulation scenario

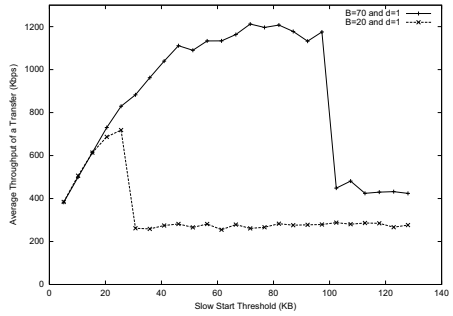


Fig. 10. Throughput vs.  $W_{th}$

21KB. We see well that the throughput starts to deteriorate approximately in the middle between 0 and these values of  $W_B^{max}$ . The biggest decrease in the performance appears exactly at  $W_B^{max}$ .

Thus, in case of multiple connections,  $W_{th}$  must be set according to equation (6). However, overestimating the overflow window doesn't lead to an important degradation in performance as long as  $W_{th}$  is set less than the maximum overflow window given in equation (7).

## 5.2 Validation of Decreasing Byte Counting

We compare here our algorithm to STCP and BC. A  $d_{max}$  equal to 2 is considered. The simulation scenario of the previous section is used. However, in this case we set  $W_{th}$  to the BDP ( $\simeq 350$  packets). We change  $B$  and we plot for each strategy, two performance measures. In Figure 11, we plot the average throughput achieved by a connection. In Figure 12, we plot the average of the ratio of the number of uniquely transmitted packets to the total number of transmitted packets. We call this average the *Retransmission Ratio*. It indicates how much aggressive is a strategy. This ratio must be as much as possible close to one.

In Figure 11, we see clearly the three regions we have talked about in section 4.3. For small buffers, STCP gives the best performance but our proposition still gives better performance than BC. For a medium  $B$ , our proposition gives the best performance. At large buffers, BC is no longer aggressive and it outperforms the two other strategies but our proposition still gives better performance than STCP. Given that it merges the two strategies, the performance of DBC is either in between or better than the two others. Note here that the buffer size at which the throughputs jump up can be easily validated using equation (6).

Concerning the number of retransmitted packets, it is clear how BC gives the largest number and how STCP gives the smallest one. This number increases with the increase in the aggressiveness with an important difference between the three strategies at small buffers. Again here, we see the three regions we talked about. The retransmission ratio of a strategy moves to one when the buffer size becomes large enough to absorb its burstiness.



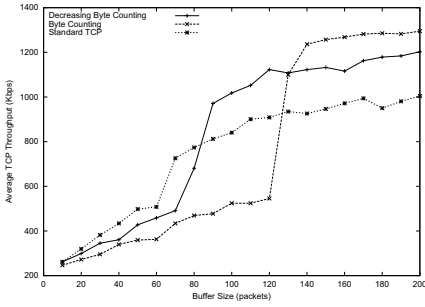


Fig. 11. Throughput vs. buffer

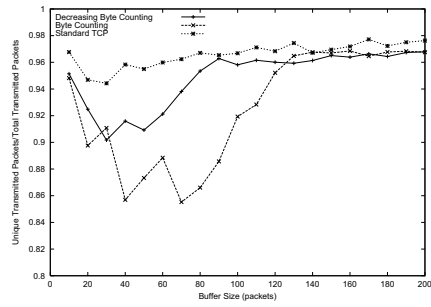


Fig. 12. The retransmission ratio

## 6 Conclusions

In this paper, we study the behavior of SS at the beginning of a connection and its impact on the performance. We calculate first the value to which the SS threshold must be set. This value can be independent of the network capacity and function only of the buffer size. We study then the impact of the window increase rate during SS. We show that accelerating the window increase improves the performance until network buffers become unable to absorb the bursts of SS. Beyond this point, any increase in SS aggressiveness deteriorates the performance. We define the optimum window increase strategy during SS and based on this, we present a new algorithm for the window increase that reduces the duration of SS while not overloading the network buffers. With this strategy, the ACK clock is preserved and the SS threshold can be always set to the estimate of the network capacity.

## References

1. M. Allman, "On the Generation and Use of TCP Acknowledgments", *Computer Communication Review*, Oct. 1998.
2. M. Allman, S. Floyd, and C. Partridge, "Increasing TCP's Initial Window", RFC 2414, Sep. 1998.
3. M. Allman, D. Glover, and L. Sanchez, "Enhancing TCP Over Satellite Channels using Standard Mechanisms", RFC 2488, Jan. 1999.
4. E. Altman, J. Bolot, P. Nain, D. Elouadghiri, M. Erramdani, P. Brown, and D. Collange, "Performance Modeling of TCP/IP in a Wide-Area Network", *34th IEEE Conference on Decision and Control*, Dec. 1995.
5. C. Barakat, E. Altman, and W. Dabbous, "On TCP Performance in a Heterogeneous Network : A Survey", *IEEE Communication Magazine*, Jan. 2000.
6. C. Barakat, N. Chaher, W. Dabbous, and E. Altman, "Improving TCP/IP over Geostationary Satellite Links", *IEEE Globecom*, Dec. 1999.
7. K. Fall and S. Floyd, "Simulation-based Comparisons of Tahoe, Reno, and SACK TCP", *Computer Communication Review*, Jul. 1996.
8. A. Kumar, "Comparative Performance Analysis of Versions of TCP in a Local Network with a Lossy Link", *IEEE/ACM Transactions on Networking*, Aug. 1998.

9. J. Hoe, "Improving the Start-up Behavior of a Congestion Control Scheme for TCP", *ACM Sigcomm*, Aug. 1996.
10. V. Jacobson, "Congestion avoidance and control", *ACM Sigcomm*, Aug. 1988.
11. T.V. Lakshman and U. Madhow, "The performance of TCP/IP for networks with high bandwidth-delay products and random loss", *IEEE/ACM Transactions on Networking*, Jun. 1997.
12. The LBNL Network Simulator, *ns*, <http://www-nrg.ee.lbl.gov/ns>.
13. W. Stevens, "TCP Slow-Start, Congestion Avoidance, Fast Retransmit, and Fast Recovery Algorithms", *RFC 2001*, Jan. 1997.

# Performance Study of Satellite-Linked Web Caches and Filtering Policies

Xiao-Yu Hu, Pablo Rodriguez \*, and Ernst W. Biersack

Institut EURECOM, Sophia Antipolis, France.  
{rodrigue,erbi}@eurecom.fr

**Abstract.** The exponential growth of World-Wide Web (WWW) requires new and more efficient content distribution mechanisms. Web caching has shown to be a very efficient way to scale the WWW by reducing traffic in congested network links, decreasing latency to the clients, and reducing load in the origin servers. In this paper we study an emerging caching strategy called cache-satellite distribution. In a cache-satellite distribution caches are inter-connected via a satellite channel, effectively increasing the client population connected to a cache. Using Markov chain analysis, we demonstrate that the higher the number of clients connected to a cache-satellite distribution, the higher the probability that a document request is hit in the cache. Our analytical results strongly advocate a large scale interconnection and cooperation of ISP-caches via a satellite distribution.

Due to the large amount of documents distributed through the satellite and the limited disk capacity and processing power of local ISP caches, it is impossible to store all documents coming from the satellite distribution. Therefore, it becomes necessary for an ISP-cache connected to the satellite distribution to perform efficient filtering policies. Based on the stability in the behavior of the clients of an ISP cache, we propose novel filtering policies which rely on the Web servers visited on the previous days. Using trace-driven simulation, we study different filtering policies and show how simple filtering policies can exhibit excellent performance in rejecting non-desired documents while assuring high hit rates.

**Key Words:** World-Wide Web, Web Caching, Satellite Distribution, Filtering Policies.

## 1 Introduction

The Web today is characterized by high volume of accesses to popular Web pages. Thus, identical copies of many documents pass through the same congested network links. As a result, network administrators see a growing utilization of their

---

\* Pablo Rodriguez is supported by the European Commission in form of a TMR (Training and Mobility for Researchers) fellowship. EURECOM's research is partially supported by its industrial partners: Ascom, Cegetel, France Telecom, Hitachi, IBM France, Motorola, Swisscom, Texas Instruments, and Thomson CSF.

network that requires upgrading their links or replacing their servers and end users experience longer and longer latencies to retrieve a document. These problems can be alleviated by widespread migration of copies of popular documents from servers to points closer to the users [2]: caching at the origin server, caching at the client (e.g., caches built into Web browsers), and most importantly, proxy caching servers inside the network, also called Web caches.

World-Wide Web caches can potentially reduce the number of requests that reach popular servers, the volume of network traffic resulting from document requests, and the latency that an end-user experiences in retrieving a document. However, caches offer limited performance since there is a large percentage of requests that are not satisfied by the cache. Requests satisfied in the cache are called *hits*. Requests not satisfied in the cache are called *misses*. Misses can be classified into: 1) *First-Access*: misses occurring when requesting documents for the first time. 2) *Capacity*: misses occurring when accessing documents previously requested but discarded from the cache due to space limitations. 3) *Updates*: misses occurring when accessing documents previously requested but already expired. 4) *Non-cacheable*: misses occurring when accessing documents that need to be delivered from the origin server (e.g. dynamic documents generated from cgi-bin scripts)

Even when a cache has an infinite storage capacity, the number of misses in an institutional cache can be very high (50 – 70%) [2]. While non-cacheable documents typically do not account for more than 10% of all requests [14] and update misses do not account for more than 9% of all requests, there is a large percentage of requests that result in first-access misses (30-50%) [14]. One way to reduce the number of first-access misses and update misses is to prefetch the cache; documents are pushed into the cache even if the cache has never requested them. The idea is to get documents in the cache expecting that clients will likely request them. When documents are prefetched through the Internet and no client requests them, the bandwidth waste is considerable. An emerging alternative to prefetch or push web documents into caches is using a satellite distribution [13] [4] [10]. A satellite distribution has fewer packet losses and congestion problems than a distribution in the Internet. Also, a satellite distribution can reach a very large population with relatively little effort; adding a new additional client does not increase the cost of transmission.

The principle of pushing popular Web documents into caches via a satellite is quite simple: every requested document that results in a miss in any ISP-cache, is automatically broadcasted by a *master distribution center* to all the other ISP caches through a satellite channel. As the number of clients in all caches interconnected by the satellite continually increases, the probability that a specific client is the first requesting a certain document steadily decreases, thereby obtaining higher hit rates. Recently several companies like Sky Cache [13] and Edgix [7] have started to offer this kind of service.

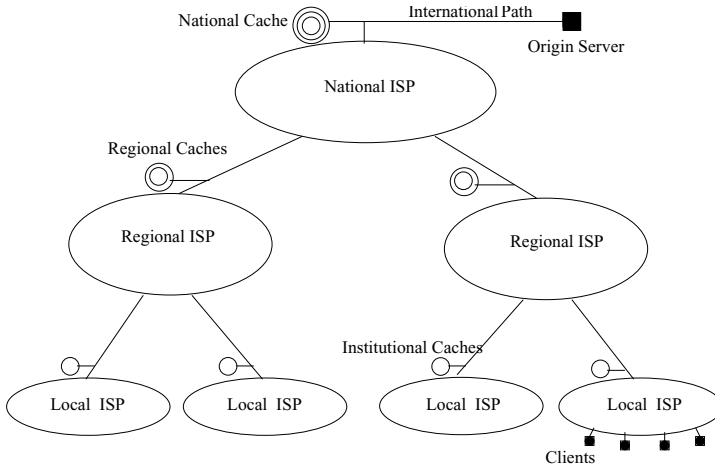
In this paper first we develop a discrete time-evolving model to describe the hit-or-miss behavior of the requested object, enabling a very simple numerical expression for the hit rate (HR). By means of this explicit HR formula, we

can easily determine the hit rate for a document as a function of the client population, and we can also study filtering policies at the master distribution center. Using trace-driven simulations we show how the hit rate of an ISP can be easily improved when it gets connected to a satellite distribution where many clients share the same caches.

Second, we study different filtering policies at the ISP-cache side. As more and more clients get connected into the satellite caching distribution, a large number of documents are to be stored at each ISP-cache, which might overflow the cache storage capacity. Therefore, there is a need for efficient filtering techniques that block non-desired documents coming from the satellite link into the ISP caches. Based on the assumption that the ISP clients' interest do not change dramatically on a day-by-day basis, we propose a novel filtering policy which relies on the Web servers visited on the previous day. The decision rule is simple: if an incoming document matches the filtering database with a list of Web servers visited on the previous day, then store the document for a possible future hit; otherwise just discard it. The philosophy behind this filtering policy is that if a Web server was visited by local clients on the previous day, then this Web server is very likely to be requested again by a local client. We also consider other filtering policies where only those documents coming from previously visited Web sites with more than a certain number of requests are considered. Using trace-driven simulation, we study different filtering policies and show that simple filtering policies can exhibit excellent performance in blocking non-desired documents while retaining high hit rates.

## 2 Internet Topology: An ISP Perspective

At its current stage, the Internet connecting the server and the clients can be modeled as a hierarchy of ISPs, with each ISP having its own autonomous administration. Without loss of generality, we can make a reasonable assumption that the Internet hierarchy consists of three levels of ISPs: local, regional, and national, as shown in Figure 1. All the clients are connected to the institutional ISP, which might serve a corporation or university. The local ISP is usually characterized by local area coverage and thus a high speed bandwidth. The local ISPs are connected to the regional networks and the regional networks are upwardly connected to the national ISP. In hierarchical caching, caches are usually placed at the access points between two different networks to reduce the cost of transmitting across a new network [5]. One popular protocol which allows Web clients to coordinate and share a hierarchy of Web caches is the Internet Caching Protocol (ICP) [15]. However, the hierarchical topology and ICP have several drawbacks: First, the disk space utilization is relatively low due to *mirroring effect*—only a fraction of the total storage capacity contains unique objects. Second, if the document is not hit in the cache, additional latency is added when searching the documents among the siblings and traversing the caching hierarchy. Third, higher levels of the caching hierarchy may easily become highly congested and add considerably delays [11].



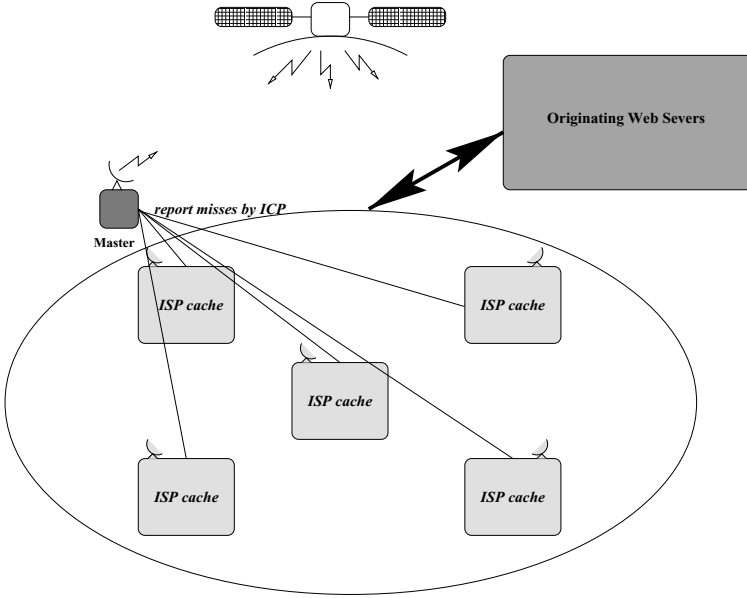
**Fig. 1.** Internet topology: hierarchical structure.

Recently, an emerging cache pushing mechanism called cache-satellite distribution is receiving more and more attention [10]. The topology structure for a cache-satellite distribution is shown in Figure 2. Since satellite links usually have broad bandwidth and cover a large scale of geographical areas, there is no need for a caching hierarchy, which aims at creating a large effective client community. Compared with hierarchical caching, a satellite distribution has the ability to cover more clients with less cost.

The satellite caching distribution works as follows. Whenever there is a miss at an ISP cache, the ISP cache obtains the document from the origin Web server via HTTP protocol. The ISP cache reports the missed URL (Universal Resource Location) to a master site via ICP [15]. The master site then obtains the document from the origin server and transmits the document into the satellite channel. As a result, all ISP caches receive the broadcasted document and can decide to keep it or not. Each ISP cache receives all the documents requested by any client connected to the satellite caching distribution, therefore, the probability that a specific client is the first client requesting a document is very small and the hit rate increases.

### 3 Analytical Model and Performance Study

In this section we present analytical models to derive an explicit hit rate expression depending on the client community connected to the cache, and the life-time of a document. Assume that we have totally  $N$  clients covered by satellite distribution and that each ISP cache has infinite disk space. Let  $\lambda_i$  be the request rate of a single user for a *cacheable* document, where  $i = 1, 2, \dots, N$ . Assume that the request rate for a single document is Poisson distributed [9],



**Fig. 2.** Satellite distribution.

therefore, the total request rate  $\lambda$  for a document is

$$\lambda = \sum_{i=1}^N \lambda_i \quad (1)$$

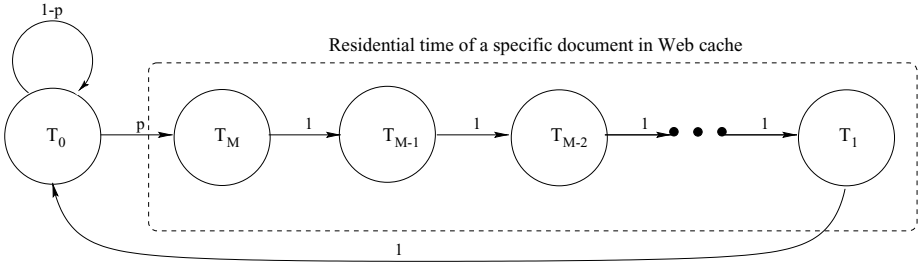
which is also a Poisson process with mean  $\lambda$ .

We define the mean life time duration of a document in a Web cache by the term *residential time*  $T_s$  which depends on several factors: estimated time-to-live (TTL), modification cycle, and filter/removing policy. We assume that the residential time is exponentially distributed with mean  $T_s$  [6].

Now, we define an arbitrary small time-slot  $\tau$ , which tends to zero, and denote  $T_M, T_{M-1}, \dots, T_1$  as the remaining residential time of a specific document in a Web cache after a time  $\tau$ , where  $M = T_s/\tau$  (see Figure 3). When the residential time  $T_s$  expires, the state  $T_0$  is entered again. Therefore, the probability to enter state  $T_M$  is given by  $p = \tau\lambda$ .

Denote the steady-state probability to be in state  $T_n$  by  $\pi(T_n)$ . From the Markov chain, we know that

$$\begin{cases} \pi(T_0) &= \pi(T_1) + (1-p)\pi(T_0) \\ \pi(T_M) &= p\pi(T_0) \\ \pi(T_{M-1}) &= \pi(T_M) \\ \pi(T_{M-2}) &= \pi(T_{M-1}) \\ &\dots \\ \pi(T_1) &= \pi(T_2) \end{cases}$$



**Fig. 3.** Markov chain model.

and furthermore  $\sum_n \pi(T_n) = 1$ .

Solving the equations above, we obtain that  $\pi(T_M) = \pi(T_{M-1}) = \pi(T_{M-2}) = \dots = \pi(T_1)$ , and  $Mp\pi(T_0) + \pi(T_0) = 1$ , which leads to

$$\pi(T_0) = \frac{1}{1 + pM} \quad (2)$$

Substituting  $M = T_s/\tau$  and  $p = \tau\lambda$  into equation 2 yields  $\pi(T_0) = \frac{1}{1 + \lambda T_s}$ . The probability that a miss occurs is the probability that a request finds a document in state  $T_0$ . Therefore the miss rate (MR) is equal to  $\pi(T_0)$ , i.e.  $MR = \pi(T_0) = \frac{1}{1 + \lambda T_s}$  and the hit rate (HR) for a given document is

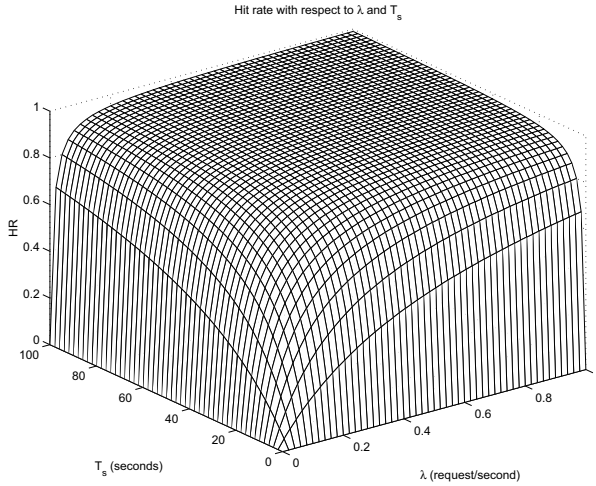
$$HR = 1 - MR = \frac{\lambda T_s}{1 + \lambda T_s} \quad (3)$$

In Figure 4 we plot the hit rate HR with respect to  $\lambda$  and  $T_s$ . From Figure 4 we see that for a specific cacheable document, as  $\lambda$  or  $T_s$  increases, the HR tends to 100%. Therefore, the larger the number of clients connected to satellite distribution, or the larger the residential time of a document  $T_s$ , the higher the probability that a document requested by a client is hit in the Web cache.

## 4 Filtering Policies

It is worth pointing out that the results obtained in equation 3 assume that the satellite bandwidth and the cache disk storage are infinity. Practically, the satellite bandwidth and the cache's storage capacity are limited, thus, filtering or removing policies are needed. One simple filtering policy at the master side could be to avoid sending those documents that are not very popular. Thus, the master site of a satellite distribution would simply set up a threshold  $TSH$  on the number of requests per update period  $\lambda T_s$  and distribute through the satellite only those documents with  $\lambda T_s \geq TSH$  requests. This, simple filtering policy at the master side can keep the overall hit rate high, while making the bandwidth and disk space requirements affordable [10].





**Fig. 4.** Hit rate with respect to  $\lambda$  and  $T_s$

At the cache side (i.e. ISP caches), it may be difficult or impossible to obtain (or estimate) the value of  $\lambda T_s$ . Also, clients at a given cache may be interested in a different set of documents than clients at a different cache, therefore, cache side filtering policies should be implemented. Good filtering policies at the cache side would require to block most of the unnecessary documents or documents that are unlikely to be hit in the future, while retaining a high hit rate. For instance, if most of our clients are college students, it might be necessary to keep all the documents related to *.edu*; and if most of our clients speak only Chinese, it may be reasonable to discard all documents coming from non-Chinese, non-English speaking country. Of course this kind of prior information is very helpful to establish an efficient filtering policy. However, in this paper we assume no such prior knowledge on the background of clients; we consider a more general case: only the history of the requests of local clients is available in the form of caching server's logs.

Compared with the fast changing rate of the Internet and its documents, the relish and interest of each member of the local users (clients) usually remains stable, at least for a short period, and so does the membership of the audience itself as a whole. Accordingly, the requesting pattern from all ISPs could also be thought to be relatively stable. Based on this assumption, we suggest a *generic filtering policy* which depends solely on the previous day's request pattern at each local site. For a specific day, we first summarize the total Web servers that were visited on the previous day and build a database with each entry representing an origin Web server. Then, for each document coming from the satellite the ISP's cache decides whether to accept it or not by comparing the host name of the document's URL with the filtering database. If there exists a match, the document is stored into the cache for possible future hits; otherwise, it is simply discarded. The philosophy behind this filtering policy is that if a Web server was

visited by local clients on the previous day, then it is very likely to be requested again by local clients.

We then extend the proposed filtering policy further to take into account the popularity of a document. Based on the popularity of a Web server, we can set up different thresholds to determine which documents to keep. We call this method a *threshold filtering policy*.

Next, we present via trace-driven simulations the performance of a generic and a threshold filtering policy in terms of disk space usage, hit rate (HR), and weighted hit rate (WHR) – the fraction of client-requested bytes returned by the web cache. We use the access logs provided by an ISP in the USA (AYE [1]). This ISP gives access to about 1000 residential and business clients and has a cache with 48 GBytes of disk space. The logs of AYE's cache are from Dec. 18-23, 1998, which account for nearly 10 million requests. We considered all documents, cacheable and uncacheable, to study different filtering policies regardless of the cacheability of a document (non-cacheable documents account for about 10% of all document requests).

## 5 Filtering Policies: Trace-Driven Simulation

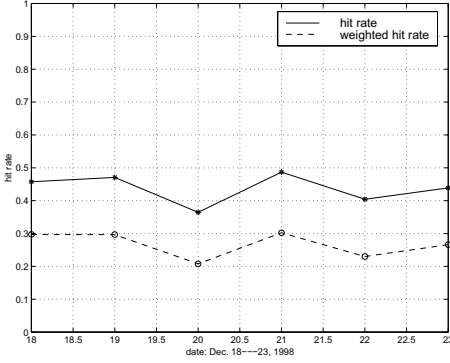
### 5.1 Original Hit Rate and Weighted Hit Rate

In order to measure the influence of the filtering policies on the hit rate, we first calculate the original hit rate without the use of the filtering policies. To calculate the hit rate at AYE's cache we considered *all* hits, including those hits that needed a consistency check (if-modified-since) with the origin server and resulted in a document not-modified [8]. From Figure 5 we can see that the average hit rate HR of AYE's cache is about 45% and the average weight hit rate WHR is 30%. Both the HR, and the WHR are quite low since most of the document requests result in a miss in the cache. Misses are mostly due to requests for documents that no other client has ever requested before (first-access misses). The significant difference between HR and WHR is because Web caches usually discard large documents to help keeping high HR [12].

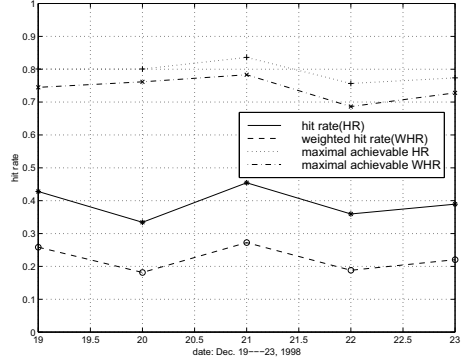
### 5.2 Impacts of a Generic Filtering Policy on the Hit Rate

In Figure 6 we consider a generic filtering policy where ISP caches only keep those documents which host name matches that of any Web site visited during the previous day. Comparing Figures 5 and 6, we see that only a quite negligible reduction in the HR and the WHR has been caused by the generic filtering policy. For example, on Dec. 19, the corresponding HR reduction is only 4.3%, on Dec. 20 only 3.0%, on Dec. 21 only 3.2%, and so forth.

In Figure 6 we also calculated the *maximal achievable hit rate* and the *maximal achievable weighted hit rate*, which is the HR and WHR achieved in an ideal scenario where all documents requested in the cache were previously prefetched in the cache. If the percentage of cacheable documents is 100% then, the maximal achievable HR equals 100%. In Figure 6 we show the maximal achievable



**Fig. 5.** The original HR and WHR of AYE's cache without the use of a filtering policy.



**Fig. 6.** HR and WHR of AYE's cache with a generic filtering policy based on those servers visited on the previous day.

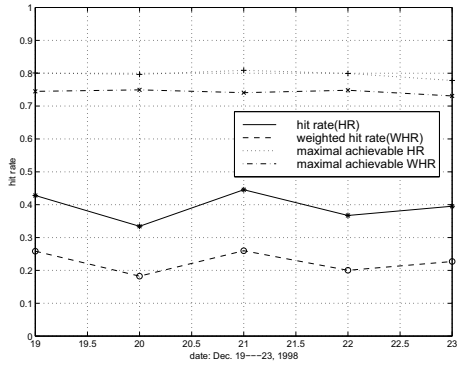
HR and WHR for the generic filtering policy described before. We observe that the maximal achievable HR and WHR is as high as 80% and 75%, respectively. Therefore, the correlation between Web sites visited from one day to the following day is very high; there are only few new Web sites that are visited on one day and are not visited on the following day. The percentage of new Web sites that are visited every day and that were not visited before only accounts for about 20% of all requests.

Not surprisingly, there currently exists a significant gap between the maximal achievable HR and the measured HR, this is due to the fact that the client population connected to AYE's cache is not large enough. However, as we will see in Section 6 as more and more ISP Web caches get connected through a satellite distribution, this gap decreases rapidly.

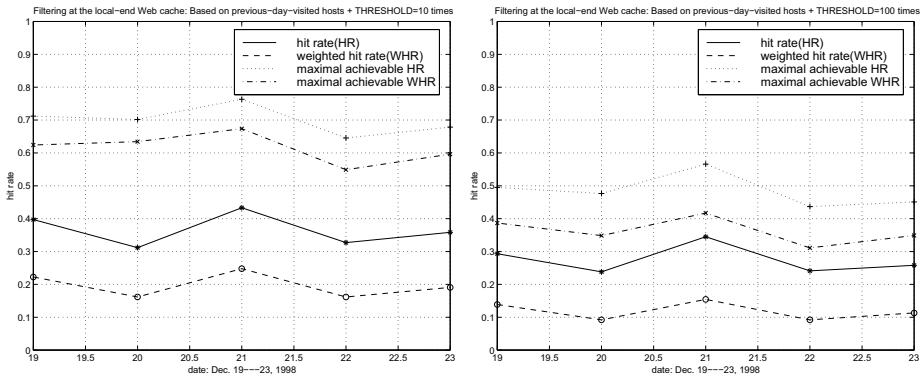
Next, we have also considered another interesting case where we employ a unique database built on Dec. 18 to implement filtering policies for Dec. 19, 20, 21, 22, and 23 respectively. Thus, during the days Dec. 19-23, the cache only keeps those documents which host name matches that of a Web site visited on Dec 18. The results, shown in Figure 7, exhibit a surprising resemblance with Figure 6 where the filtering databases were built based on the immediate previous day. This results, strongly support the fact that the relish and interest of clients connected to an ISP remain stable on a day-by-day basis.

### 5.3 Performance of a Threshold Filtering Policy

In this section we consider a threshold filtering policy. The only difference between a threshold filtering policy with the previous one (generic) is that all the entries in the database from a Web site with less than a certain number of requests during the previous day (i.e. threshold) are removed. That is to say, if a Web server was visited less than threshold times on the previous day, it means



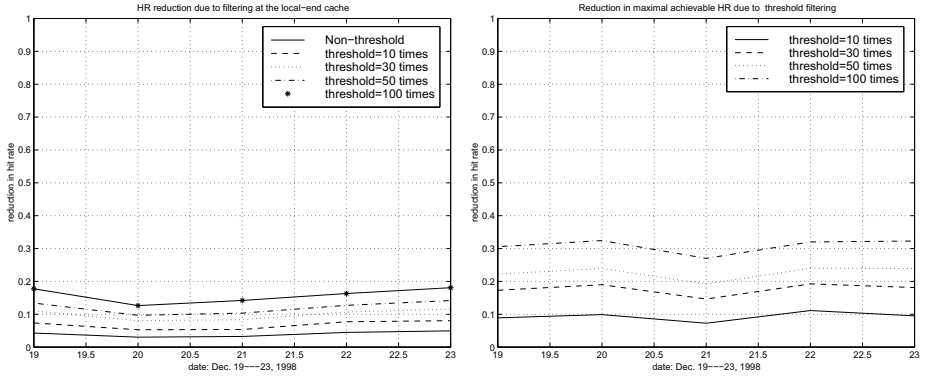
**Fig. 7.** HR and WHR of AYE's cache with filtering policy based on servers visited on a specific day: Dec. 18.



**Fig. 8.** Performance of threshold filtering policy with thresholds equal 10 req/day, and 100 req/day, respectively.

this server is not a “hot” site, and therefore the cache decides not to store the documents coming from it. In this way, we can readily reduce the number of entries used to implement the filtering policy, thereby, alleviating the demands for processing power and disk space. However, reducing the number of Web sites for which the cache keeps documents may decrease the hit rate. The goal is to reduce the number of Web sites for which the cache keeps documents while retaining the HR and the WHR at an acceptable level.

In Figure 8 we show the HR and the WHR for different threshold filtering policies based on previous day logs. Moreover, in Figure 9 we also plot the reduction in HR and WHR of each threshold filtering policy, when compared to the non-filtering case. The results in both figures show that setting a small threshold popularity, does not significantly decrease the hit rate. For instance, setting a threshold of 10 requests per day, the hit rate is only decreased by about 3% compared to the case where no threshold is used. The main reason for this



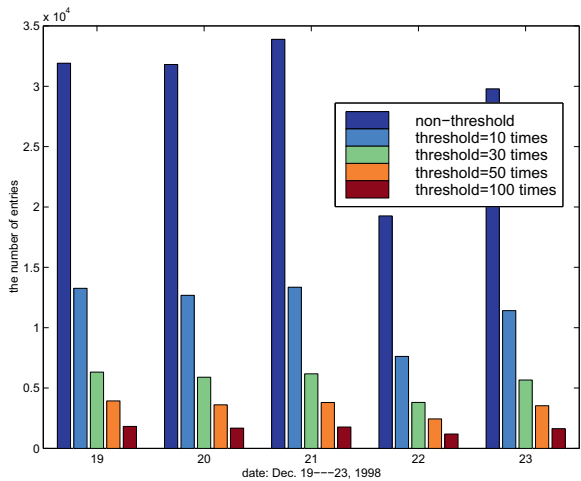
**Fig. 9.** Comparison of HR and WHR reduction due to different filtering policies.

results from the Zipf distribution of document requests [3] [16] where only few Web sites account for most of the requests and there is a large set of documents that have very few requests.

#### 5.4 Complexity and Disk Requirements of Filtering Policies

Up to now we have considered the performance of generic filtering policies based on previous requesting patterns and threshold filtering policies with different thresholds. As we saw, the mentioned filtering policies only have a negligible (or tolerable) reduction in both HR and WHR. However, other issues arise with filtering policies, such as the computing requirements for executing such filtering policies, and the disk requirements to store the documents coming from satellite link depending on the filtering policies.

For each document coming from the satellite, it is necessary for a filtering policy to examine whether the document's host name matches one of the filtering database entries. Therefore, the execution complexity of the filtering policy depends on the number of entries in the database. Also, since all the documents with a host name equal to an entry in the database are stored in the cache, the number of entries in the database is directly related to the cache disk requirements. The more entries in the filtering database, the more the storage space is required. In Figure 10 we plot the number of entries needed for the proposed filtering policies. From Figure 10 we see that the number of entries decreases quickly as the threshold value increases, resulting in a drastic reduction in the execution complexity and the needed disk space. At the same time, from Figures 8 and 9 we also saw that the decrease in the hit rate was small even for large threshold values. For instance, on Dec. 19, the total number of entries used for a generic filtering policy (i.e. the non-threshold case) is 31913, and the hit rate is 42.8%; the threshold filtering policy with threshold=10 req/day uses 13266 entries, less than half of the generic policy, while still achieving a hit rate of



**Fig. 10.** The number of entries used in different filtering policies.

39.7%. Therefore, implementing threshold filtering policies helps reducing the execution complexity and disk requirements while keeping high hit rates.

**5.5 Filtering Policy Based on Consecutive Days of Access History**

Next, we investigate how many days of access history are needed to build the filtering database. Obviously, using more days of history, the probability to hit a document will increase at the cost of a large filtering database and processing power. In Table 1 and 2 we present the increased hit rate and increased number of entries, respectively, as compared to those of filtering policy based solely on the previous day. We only considered the generic filtering policy, i.e., the threshold is set to zero.

From these two tables we see that increasing the number of days used to build the database, rapidly increases the number of database entries, while, it does not significantly improve the hit rate. For example, on Dec. 20 the filtering policy based on the two previous days has a 1.2% higher hit rate than that based only on the previous day, however, the number of entries in filtering database grows

**Table 1.** Increased hit rate based on consecutive days of history in the case of a generic filtering policy.

filtering database based on previous ...	increased HR on Dec. 20	increased HR on Dec. 21	increased HR on Dec. 22	increased HR on Dec. 23
two days	1.2%	1.2%	2.6%	1.4%
three days	N/A	1.7%	3.1%	2.9%
four days	N/A	N/A	3.4%	3.5%
five days	N/A	N/A	N/A	3.8%

**Table 2.** Additional number of entries for consecutive days of history in the case of generic filtering policy.

filtering database based on previous ...	increased entries for Dec. 20	increased entries for Dec. 21	increased entries for Dec. 22	increased entries for Dec. 23
two days	16995(53.4%)	16498(48.7%)	27389(142.2%)	9264(31.1%)
three days	N/A	29130(85.9%)	37981(197.2%)	26411(88.6%)
four days	N/A	N/A	49621(257.6%)	38678(129.8%)
five days	N/A	N/A	N/A	48769(163.7%)

by 16995 items, which is a 53.4% more. As more days of history are employed, the increased hit rate is almost negligible, while the increase in the number of filtering entries is quite significant. Therefore, using only the previous day of logs to build the filtering database is a reasonable choice.

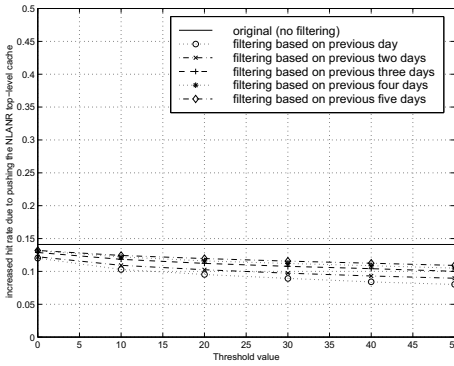
## 6 Case Study: Filtering Schemes on a Cache-Satellite Distribution

Now, we proceed to study the situation where AYE's cache is connected to a cache-satellite distribution. As shown in Section 3, as more and more clients get connected to the cache-satellite distribution, the hit rate increases gradually, however, the disk space requirements will increase explosively if no filtering policy is employed. The primary goal of this section is to verify that the filtering policies presented before have the ability to keep most of the documents coming from the satellite that are of interest to the local clients, while blocking as many non-desired documents as possible.

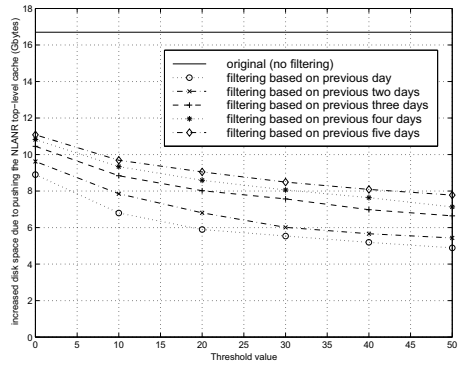
For this purpose we obtain one day of logs from the top-level log of NLNR<sup>1</sup> cache on Dec. 23 and simulate a scenario where the NLNR top-level cache gets connected to cache-satellite distribution. We assume that AYE's cache is connected to the same satellite distribution. We consider that all objects requested in the NLNR log, whether hit or missed, are pushed over the satellite link and therefore received by AYE's cache. We are interested on the increased hit rate at AYE's cache due to satellite pushing, and on the additional required disk space at AYE's cache.

In Figure 11 and Figure 12 we show the percentage increase of the hit rate for different filtering policies and the increase of the disk requirements. From these two figures we see that in the case that no filter is used, an additional 16.7 GBytes of disk space are required to store those documents coming from the satellite at AYE's cache, and the resulting hit rate increases by 14.1%. Using the generic filtering algorithm, i.e., keeping only the incoming documents that match Web servers visited on the previous day, the additional disk space requirement drops to 8.9 Gbytes while the increased hit rate is still about 12.0%. Therefore, the generic filtering policy achieves 46.2% disk-space saving at the cost of a negligible decrease in HR.

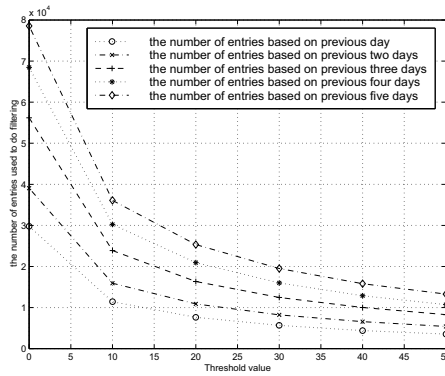
<sup>1</sup> National Lab of Applied Network Research. <http://ircache.nlanr.net>. NLNR top-level cache consists of four major cooperating caches that share their load.



**Fig. 11.** The original HR and WHR of AYE's cache without the use of a filtering policy.



**Fig. 12.** HR and WHR of AYE's cache with a generic filtering policy based on those servers visited on the previous day.



**Fig. 13.** Number of entries in the database used to filter documents coming from the satellite.

We further study the increased hit rate corresponding to threshold filtering policies based on several previous days. As shown in Figure 11, we find that, as more and more days' logs are incorporated into filtering database, the increase in the hit rate is quite negligible. On the other hand, in Figure 12 we see that as more days are considered, the needed disk space to store those documents coming from the satellite is significantly enlarged; a similar thing happens to the number of database entries (Figure 13). For instance, using the previous day of logs from AYE, the additional hit rate increases by 12.0%, AYE's cache needs additionally 8.9 Gbytes of disk space, and the database has 29793 filtering entries; based on the previous two days of logs, the increase in the hit rate is of 12.5%, however, the increase in disk space is 9.62 Gbytes and the number of entries is 39058, and so forth. Based on these figures, we find that the use of a generic filtering policy or a threshold filtering policy with small threshold values, e.g. 10 req/day,



based solely on the previous day, yields to a good trade-off between disk space requirements to store Web documents, processing capacity to run the filtering policies, and hit rate.

## 7 Conclusions

A cache-satellite distribution is emerging as a very promising technology to alleviate the problems related to the rapid growth of the Web. In this paper, we have analyzed and evaluated the performance of cache-satellite distribution. Our theoretical and trace-driven results demonstrate that as the population of clients connected to the satellite distribution increases, the hit rate goes up steadily. Using trace-driven simulations we showed that connecting more clients to the satellite distribution scheme, ISP-caches can easily increase their hit rates by 15%.

As more and more clients (or ISPs) get connected to a cache-satellite distribution, a large number of documents is being distributed through the satellite, which might overflow the disk storage capacity of the caches and considerably increase their load. We have studied different filtering policies at the ISP caches, aiming at discarding documents unlikely to be requested. In particular we have considered filtering policies which take into account access patterns from clients connected to a certain cache. Using trace driven simulations, we show that simple filtering policies can greatly reduce the disk requirements of Web caches connected to a satellite distribution while reducing the hit rates only by about 2%-3%. In addition, filtering policies that include the site's popularity to decide whether to keep a document or not, further reduce the size of the filtering database while hardly modifying the obtained hit rates.

## References

1. AYE, "http://www.aye.net/".
2. M. Baentsch, L. Baum, G. Molter, S. Rothkugel, and P. Sturm, "World Wide Web Caching: The Application-Level View of the Internet", *IEEE Communications Magazine*, pp. 170-178, June 1997.
3. L. Breslau, P. Cao, L. Fan, G. Phillips, and S. Shenker, "On the Implications of Zipf's Law for Web Caching", In *Proceedings of IEEE INFOCOM'99*, New York, USA, March 1999.
4. Broadcast Satellite Services, "http://www.isp-sat.com".
5. A. Chankhunthod et al., "A Hierarchical Internet Object Cache", In *Proc. 1996 USENIX Technical Conference*, San Diego, CA, January 1996.
6. F. Douglass, A. Feldmann, B. Krishnamurthy, and J. Mogul, "Rate of change and other metrics: A live study of the World Wide Web", In *Proceedings of the USENIX Symposium on Internet Technologies and Systems*, December 1997.
7. Edgix, "http://www.edgix.com".
8. R. Fielding, J. Gettys, J. Mogul, H. Frystyk, T. Berners-Lee, et al., "RFC 2068: Hypertext Transfer Protocol — HTTP/1.1", January 1997.

9. S. Gribble and E. Brewer, "System Design Issues for Internet Middleware Services: Deductions from a Large Client Trace", In *Proceedings of the USENIX Symposium on Internet Technologies and Systems*, December 1997.
10. P. Rodriguez and E. W. Biersack, "Bringing the Web to the Network Edge: Large Caches and Satellite Distribution", In *To appear in MONET. Special issue on Satellite-based information services*, January 2000.
11. P. Rodriguez, K. W. Ross, and E. W. Biersack, "Distributing Frequently-Changing Documents in the Web: Multicasting or Hierarchical Caching", *Computer Networks and ISDN Systems. Selected Papers of the 3rd International Caching Workshop*, pp. 2223–2245, 1998.
12. A. Rousskov, "On Performance of Caching Proxies", In *ACM SIGMETRICS*, Madison, USA, September 1998.
13. SkyCache, "<http://www.skycache.com>".
14. R. Tewari, M. Dahlin, H. M. Vin, and J. S. Kay, "Beyond Hierarchies: Design Considerations for Distributed Caching on the Internet", In *Proceedings of the ICDCS '99 conference*, Austin, Texas, May 1999.
15. D. Wessels and K. Claffy, "Application of Internet Cache Protocol (ICP), version 2", Internet Draft: draft-wessels-icp-v2-appl-00. Work in Progress., Internet Engineering Task Force, May 1997.
16. G. K. Zipf, *Human Behavior and the Principle of Least Effort: An Introduction to Human Ecology*, Addison-Wesley, Reading, MA, 1949.

# **Distributed Control System for Low Priority Controllable Traffic in Packet Switched Backbone Networks**

Kimmo Pulakka and Jarmo Harju

Tampere University of Technology  
Telecommunications Laboratory  
P.O.BOX 553  
FIN-33101 Tampere  
FINLAND  
Tel. +358-3-365 2111  
Fax. +358-3-365 3808

**Abstract.** This paper studies the problems of congestion and optimal use of resources in packet switched backbone networks. The developed control system adjusts the data rates and selects the optimal paths through the backbone network for the controllable traffic category. Controllable traffic is low priority traffic, whose sources can be effectively controlled by the network. Data rates of the controllable traffic are calculated by the fuzzy controllers and statistical calculations. The adaptive neuro-fuzzy inference system (ANFIS) has been used to predict data rates of high priority traffic. This prediction tries to optimise the performance of the control system. All computationally intensive control operations are distributed to the edge switches of the backbone network. A special control protocol transfers the control information between the edge switches and the source stations of the controllable traffic, located in the access networks. Simulation tests have been used to verify the high performance of the control system.

## **1. Introduction**

Packet-based backbone data networks, also called core networks, transfer data between different types of access networks connected to the core by edge switches. Usually backbone networks operate in wide geographical areas and the networks consist of a large number of switches.

Different user applications require different transmission services. Hence, data transmission services should be classified according to these various needs, and backbone networks should be capable of serving data transfer requests of different service categories [1]. Currently, the idea of integrated services in packet switched networks is defined most completely in ATM (Asynchronous transfer mode) networks. ATM networks include several service categories and they are capable of sharing their transmission resources between connections according to the QoS (Quality of Service) requirements of categorised connections.

The different service categories can be roughly divided into prioritised, controllable, and best effort services. The data flows of the prioritised services are served by giving certain privileges over the other traffic in the network. The prioritised services can also include guaranteed services. For such services, the network guarantees a certain quality of the transmission services for these traffic flows. The prioritised services require transmission contracts between the user application and the network.

The controllable and the best effort traffic can use the free capacity of the network, left over by the prioritised traffic flows. The difference between the controllable and best effort service categories is the control strategy of these traffic flows. The data flows of the controllable service categories are always somehow controlled by the network so that the free capacity would be used for these traffic flows as optimally as possible. Best effort traffic is not controlled in any way.

According to the aim of controllable traffic, the traffic control systems should ensure that the controllable traffic flows are controlled in real-time according to the abrupt load changes of the backbone network. The long transmission delays of large backbone networks require some kind of prediction of the load of the network [2]. Although the prediction of the traffic load in large networks is very difficult, it is valuable to try to produce some estimations. Good predictions increase the performance of the control system and decrease the need of control actions.

In this paper, we present our own data rate control and path selection system for generic controllable (GC) traffic in packet switched backbone networks. During the data transmission periods of the controllable traffic flows, the control system iteratively chooses the optimal path and calculates the optimal data rate for each GC traffic flow, depending on the load of the higher priority traffic flows. The control system is based on the use of 1) *mamdani fuzzy controllers*, 2) *ANFIS neuro-fuzzy system* and 3) *statistical calculations*. Fuzzy controllers and statistical calculations perform the control actions for the GC sources, and ANFIS generates the estimations of the high priority traffic load values. These estimation values are used for the control of the GC data flows. Several research groups have also used fuzzy controllers for network control problems. In these studies, fuzzy controllers have been used for policing the ATM network and to control the data rates of ABR connections [3], [4], [5]. Some research groups (e.g. [6]) have also found that fuzzy control systems are also suitable for solving routing problems of networks. The ANFIS neuro-fuzzy system [7] has been found to be effective in modelling the behaviour of highly non-linear dynamic systems [8].

Our backbone network model does not follow any existing standardised network architecture. However, the data rate control of the GC traffic closely follows the basic control principles of the ABR (Available Bit Rate) service category traffic in the ATM networks, and the modelled backbone network contains typical elements of connection oriented high speed packet switched networks.

The performance of the control system has been tested by simulations. The simulation results show that the control system can reach a high level of performance, even if the physical network environment and the load of the backbone vary greatly.

In Section 2 we explain the basic properties of the control protocol, focusing on the data rate adjustment and the path selection mechanisms of the control system. In Section 3 simulation results are given and analysed. Finally, some conclusions are drawn.

## 2. Description of the Control System

### 2.1. General Principles

The studied backbone network model consists of edge and core switches. The access networks are connected to the backbone network via edge switches. The data packets coming from the source access networks flow across the source edge switches and travel through the core switches to the destination edge switches. The destination edge switches send data packets to the destination access networks. In this model, we assume that the access networks connected to the backbone network are also some kind of packet based data networks, and they can transfer all incoming and outgoing traffic without the danger of congestion. In the backbone network, data are transferred using constant length packets. In the switches, every input and output link, connected to the switches, is handled by individual input and output ports. Output ports have a scheduling mechanism for the control traffic, prioritised traffic and GC traffic (see Fig. 1). The control traffic has the highest priority and the GC traffic has the lowest priority. Best effort traffic is not included here, but if it were, its priority would be the lowest. In this model, we assume that traffic coming from input ports could be switched to the output ports without any need of input buffering. The structure of the studied network environment is described in Fig. 1.

The control system can be divided to 1) *the transmission of the control information between the network components* and 2) *the control calculations*. The transmission of the control information can be further divided to transmission between the edge switches and GC source stations, and the collection of the network state information. The control calculations include the calculations for GC data flows and the calculations to define the state of the output links of the core switches.

The edge switches periodically control the GC source stations by the special GC control (GCC) packets. These packets include the data rate values of the GC data flows to reach the optimal use of the network resources. The frequency of sending the control packets depends on the estimated mean RTT (Round Trip Time) of the network. The data rate values are calculated by the control functions of the edge stations.

The collection of the state information of the network is done by the ERM (Edge Resource Management) packets. The ERM packets, originated by an edge switch, are broadcast by the core switches of the backbone network to all other edge switches. In this way, the ERM packets flow between the edge switches using all possible paths in the network and collect path specific information about the network, roughly in the same way as the BGP protocol is used in inter-domain routing. The state information of the core switches is saved to the ERM packets according to the max-min fairness principle [9]. In this way, the core switch whose current load of the output link is the highest, is allowed to represent the state of the path. The frequency of sending the ERM packets is constant and it depends on the mean RTT of the network. After receiving any ERM packet, the edge switches update the state information related to the path of the received ERM packet to a special database.

The core switches periodically calculate state information for each output link. This state information is calculated for the ERM packets. The variables used to define states of the output links are described in Table 1. The edge switches are responsible for most of the control calculations for the GC data flows. Using the state information

of the special database of paths, they 1) *estimate the load of the high priority traffic*, 2) *define the optimal path for each active GC data flow*, and 3) *adjust the data rate values of the GC data flows to the optimal values*. Calculations of the optimal paths and optimal data rates are done periodically together with the sending of the GCC packets to the GC source stations. The optimal data rate values are copied to the GCC packets so that the GC source stations can adjust their data rates. The optimal paths of active GC flows are saved into the edge switches. Between two control moments, the edge switches direct the incoming data packets from the GC source stations to the optimal paths. The estimations of the load of the high priority traffic are made every time when edge switches receive any ERM packet. These estimations are done individually for every alternative path, and they are done by the load estimation values of ERM packets or ANFIS estimators. The estimated loads of the high priority traffic are used to define optimal paths and optimal data rates.

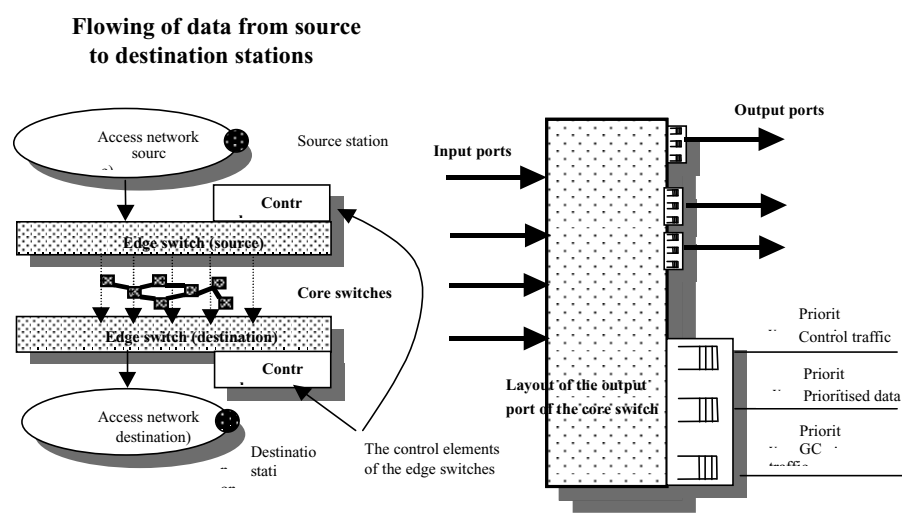


Fig. 1. Structure of studied network environment.

Table 1. Control information of core switches

Variables	Meaning
Load	Current and predicted load of the output port normalised to the link capacity. Simple linear load predictor estimates the load value of the link after RTT (round Trip Time). Load values are calculated for all traffic types.
# of GCs	Number of active GC flows through the output port
Occupancy of GC buffer	Occupancy level of the GC buffer
Transmission speed of link	The total transmission capacity of the link (for all types of traffic)

$$\text{Total.curr. load} = \sum_{i=1}^N \frac{\text{Data flow of traffic class (i)}}{\text{Capacity of the link (Mbit / s)}} \quad (1)$$

N = Number of diff. traffic classes (3)  
 {control traffic, high priority traffic, GC traffic}

$$\text{Occu.level} = \frac{\# \text{ of packets in GC buffer (packets)}}{\text{Optimal \# of packets in buffer}} \quad (2)$$

## 2.2. Prediction of the High Priority Traffic Loads

The estimation of the amount of the high priority (HIP) traffic along the paths is done in two alternative ways. If the occupancies of the GC output buffers of the heaviest loaded output links are exceeding the defined limit, the load prediction values of the latest ERM packets are used to estimate high priority loads in the paths in near future. On the other hand, if the occupancy is below this limit, the prediction is based on the ANFIS predictor. The load predictors of core switches predict the load values based on the latest load information, whereas ANFIS predictors use a little bit earlier load values for the prediction. For this reason, the prediction values of the core switches are used when congestion may occur in the path.

The ANFIS network is adaptive network based fuzzy inference system, which can perform the functions of the Sugeno or Tsukamoto fuzzy models. The important characteristic of the network is the capability to learn the behaviour of the dynamic systems by training data pairs. The network includes a set of linear and non-linear parameters. The values of the parameters are adjusted by a special hybrid learning rule, which is a combination of the gradient method and the least squares estimate (LSE). The ANFIS network can include several input variables but only one output variable [7].

The aim of the ANFIS predictors is to produce statistically reliable estimations of HIP loads for the future. Based on HIP load values measured in the past. The values of the input variables are defined by the HIP load information of several ERM packets. The input variables of ANFIS predictor are 1) *Maximum HIP load value during the observation time*, 2) *weighted average value of three latest HIP load values*, and 3) *the load change between two latest HIP load values*. The output of the ANFIS predictor is the estimation of the maximum possible HIP load value in future. The estimation time of future maximum HIP load values is defined by the operator of the control system. The ANFIS predictor is used in an on-line manner. Whenever ERM packet is received, the values of the input variables are calculated and the ANFIS network calculates the HIP load estimation for the current path. Calculated HIP load estimation values are saved into the edge switch to be used for the path selection and the data rate adjusting functions.

In our prediction case, the parameters of the ANFIS network are first adjusted in off-line mode using measured learning data pairs. The off-line mode is used, because adjusting the parameters is too time consuming to be made on-line by the edge switches. The data pairs for the parameter adjusting are collected from all possible paths in the network. In this way, the quality of the learning data pairs increases.

### 2.3. Selection of Active Path

The special databases of the edge switches contain the control information of alternative paths in the backbone network between the edge switches. The edge switches calculate the quality value for every alternative path. The paths with the best quality values will be chosen as the active paths of the GC sources. The calculation is based on two control variables. These variables are 1) *The predicted total load of the heaviest loaded output link in the path*, and 2) *the fuzzy control factor of the path*. The second variable indicates how much the data rates should be increased or decreased to reach the optimal load of the network. The values of the second variable are calculated by the fuzzy controllers. The fuzzy controller will be defined in the subsection 2.4.

The quality values of the paths are calculated according to Eq. 4. The quality of the path is high when the value of Eq. 3 is high. High values are reached when the predicted load of the link is low and the value of the fuzzy control variable is low. Low values of the fuzzy control variable indicate that the data rates of the GC traffic flows should be increased.

$$\text{path quality} = (1 - \text{PredTotLoad}) * (1 - \text{FCF}) \quad (3)$$

where :

PredTotLoad = The predicted total load of the  
heaviest loaded output link

FCF = The fuzzy control factor [0,1]

### 2.4. Data Rate Adjustments of GC Traffic Flows

**Fuzzy controller and load prediction based data rate adjustment strategies.** The data rates of the GC traffic flows are adjusted using either the fuzzy controllers or the load predictors, depending on the states of the paths of the network. Fuzzy controllers are used, when the GC data rates can be controlled without danger of congestion in the output links of the paths. On the other hand, data rate changes are based on the load predictions when there is urgently need to adapt to the abruptly high load changes of the high priority traffic.

The selection possibility of the data rate adjusting strategy tries to ensure that the occupancy levels of the GC buffers in the heaviest loaded output links in the chosen paths would always be non-zero, yet lower than the overflow level of the GC buffers. The selection procedure estimates the occupancy changes of the GC buffers during the mean RTT of the control actions (see Eq. 4). If the estimated occupancy level changes cause a buffer overflow or underflow situations, the load prediction based strategy is chosen. Otherwise, the fuzzy controller based strategy will be chosen. The selection is made individually for every path of the network in every control moment of the paths.

$$\text{Occu.change} = \frac{(\text{PredTotLoad} - 1) * \text{LC (bit / s)}}{\text{Packet size (bit)}} * \text{RTT(s)} \quad (4)$$

where :

PredTotLoad = The predicted total load of the output link

LC = Link capacity

RTT = Round trip time



**Fuzzy controller based data rate adjustments.** The fuzzy controller based data rate adjusting (see Eq. 5) is based on two control factors. These factors are 1) *the fuzzy control factor*, and 2) *the physical data rate change factor*. The fuzzy control factor indicates how much the data rates should be relatively increased or decreased normalized to the value of the physical data rate change factor. The values of the physical data rate factor indicate the allowed data rate change during one control interval. The value definition of the physical data rate change factor depends on the capacity of the network components and the geographical size of the network. (see Eq. 7).

The fuzzy controllers, designed to calculate the fuzzy control factor values, include two input variables, 1) *the occupancy of the GC buffer*, and 2) *the predicted change of the occupancy level of the GC buffer*. The input variables of the fuzzy controllers are defined in Eq. 6. The second input variable indicates how the occupancy level of the GC buffer will be changed depending on the predicted total load, during some precisely defined time period. The predicted total load is the sum of the predicted load of the control, high priority and GC data flows. The precisely defined time is usually related to the RTT of the control loop.

To ensure the simplicity of the fuzzy controllers, the fuzzy variables and the rule-bases of the fuzzy controllers are defined by the operator of the network. The first input variable is defined by seven linguistic values and second input variable by nine linguistic values. The output variable is defined by seven linguistic values and the numerical range of the variable is [0,1]. The mamdani fuzzy inference system has been used for every fuzzy controller and the centroid area method has been used for defuzzification of the fuzzy output set.

$$DR(n+1) = DR(n) + FCF(n+1) * PhyDRCF(n+1) \quad (5)$$

where :

DR = Data rate

FCF = Fuzzy control factor ( scaled to range [-1,1] )

PhyDRCF = Physical data rate change factor

$$Input1 = \frac{\text{Occupancy of GC buffer (packet)}}{\text{Optimal GC occupancy (packet)}} \quad (6)$$

$$Input2 = \frac{(\text{Pred. total load} - \text{Link capacity}) * \text{Obs. time}}{\text{Optimal GC occupancy (packets)}}$$

Optimal GC occupancy :

= Optimal # of GC data packets in the GC buffer

Obs.time = Observation time, which is  $\leq$  RTT

$$\text{PhyDRCF} = \frac{\text{Occupancy limit}}{\text{RTT}} \quad (7)$$

Occupancy limit:

Maximum allowed variance of the occupancy of the GC buffers during defined time.

RTT:

Estimated avg. round trip time of the control information

**Load prediction based data rate adjustments.** This data rate adjusting strategy is purely based on the predicted load values of different types of data flows in near future. The load values of the high priority traffic are predicted by the linear or ANFIS predictors, described in subsection 2.2. The load changes of the GC traffic flows are estimated based on the previous control actions. It is important to notice, that the predictions of the GC load changes cannot be exact values, because edge switches do not know the GC control actions of other edge switches. Because the frequency of the sending of the ERM packets is constant, the up-to-date load value of the control traffic can be used as an estimation of the control traffic load in future. Using these three predicted load values, the equally shared free capacity for the GC data flows are calculated as described in Eq 8.

$$\text{EQS capacity} = \frac{\text{Link capacity} - (\text{Pred.CL} + \text{Pred.HIPL} + \text{Curr.GCL} + \text{Pred. GCLC})}{\# \text{ of the active GC traffic flows}} \quad (8)$$

Where :

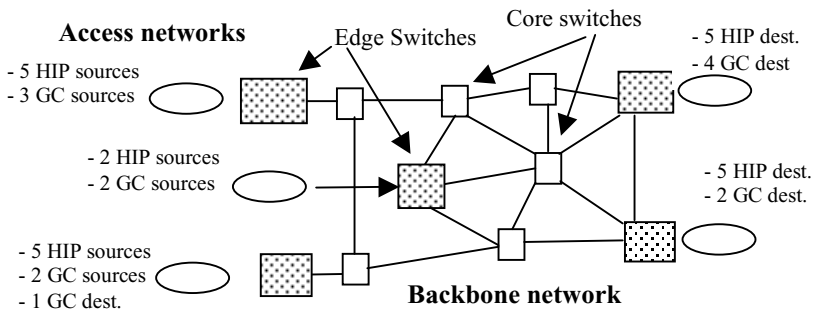
EQS capacity = Equally shared free capacity  
 Pred.CL = Predicted load of control traffic  
 Pred.HIPL = Predicted load of high priority traffic  
 Curr.GCL = Current load of GC traffic  
 Pred.GCLC = Predicted load change of GC traffic

### 3. Simulations

#### 3.1. Description of the Simulated Network Environment

The simulated network environment consisted of one backbone network and five access networks. The capacity of the transmission links between the switches of the backbone network was 35 Mbit/s, and the transmission capacity of the access networks was 100 Mbit/s. The high transmission capacities of the access networks ensure that no congestion can occur in the access networks in any case. The backbone network was loaded by 11 continuously flowing high priority traffic flows and seven GC traffic flows with randomly alternating idle and active periods. Poisson and exponential distributions were used to define the lengths of the active and idle periods of the GC traffic flows. The feasible mean lengths of the active periods were 5,10 and

20 seconds, and the mean lengths of the idle periods were 4,8, and 12 seconds. Transmission rates of seven high priority traffic flows were generated by a first-order autoregressive process described in [10]. The other four high priority traffic flows adjusted their data rates according to the measured sample data rates of H.261 video transmission. The data rates of the H.261 sources were scaled to the same data-rate range as the data-rates generated by the first-order autoregressive process. The constant data packet size in the backbone network was 500 bytes. The control system tried to share the free capacity equally between the GC traffic flows so that the network would have been loaded optimally. Another control target was to prevent congestion in the network. However, small data packet loss ratios were allowed to reach the optimal throughput. The control frequencies of the GC source stations varied according to the estimated RTT of the network. The constant interval for sending the ERM packets and measuring the loads of the output ports was 0.01 s. The structure of the simulated network environment is described in Fig. 2.



**Fig. 2.** Structure of the simulated network environment.

The transmission delays of the links and the buffer sizes of the GC buffers of the output ports in the switches will have an effect to the performance of the control system. We used four tests to study the effects of these factors. The tests differed from each other by the distances between the switches and the buffer sizes of the GC traffic. In every test, the performance of our control system was tested by five different test cases. Slightly different traffic scenarios were used for different test cases. The simulation time of each test case was 50 seconds.

### 3.2. Description of Simulation Results

**The performance variables.** The performance of the control system is tested by four performance variables. These variables are: 1) *the average occupancy of the GC buffers in the output ports of the edge switches*, 2) *the standard deviation of the occupancy of these GC buffers*, 3) *the packet loss ratio*, and 4) *mean throughput*. The values of the mean throughput were obtained by calculating, for each GC traffic flow, the ratio of the achieved throughput to the theoretical throughput value given by the max-min fairness principle under the assumption that 100% of the available capacity of the network was used.

The combination of the high throughput values and zero packet loss indicate the high performance of the system. The values of the average occupancy levels and standard deviations of the GC buffers indicate how the buffers of the output ports are used. In our simulated network environment, we have measured the performance values for the output buffers of the edge switches. These output ports should always be used heavily, because all incoming GC traffic injected to the backbone network is transferred to core switches using these output ports. The single output port is used optimally, if the average value of the occupancy level of the GC buffer is between zero and the optimal size of the GC buffer, and the value of the standard deviation is low. In our control system, the low values of the standard deviation can not be reached. The control system adaptively changes the paths of the GC traffic flows, and this causes quite high variation of the occupancy levels in the output buffers of the switches.

**Discussion about the performance results.** According to the test results, the average of the throughput values of the tests was 0.97. The variation of the throughput values of different tests was low. However, the packet loss ratio of the GC packets varies between the tests. The packet loss ratio was zero, when the optimal sizes of the GC output buffers were high enough compared to the distances between the switches. On the other hand, the packet loss ratio increases when the distances between the switches are increased, but the optimal GC buffer sizes are kept unchanged. From this point of view, it can be noticed that the packet losses can be avoided, if the sizes of the GC buffers are high enough compared to the RTT of the network. According to the average and standard deviation values of the output buffers of the edge switches, the GC output buffers have had most of the time some data packets waiting for the transmission. The test results are described in Table 2.

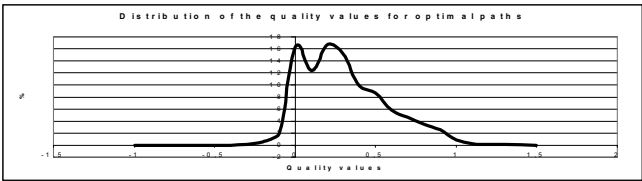
It is important to notice that the performance results (e.g. throughput values, packet loss ratios) are dependent on several factors of the control system. These factors are 1) *the frequency of the sending of the ERM packets*, 2) *the frequency of the load measurements of the output ports of the core switches*, 3) *the frequency of the sending of the GCC packet to the GC source stations*, and 4) *accuracy of the predicted load values*. Increasing the frequencies of the sending of the control information in the network and other control related measurements increases the performance of the control system, but at the same time the free capacity for the data flows decrease. On the other hand, the accuracy of the load predictions increases by more complex predictor systems, but more complex prediction system requires also more calculations. The ANFIS network used for these tests included six membership functions for each input variable and total number of fuzzy rules was 216.

**Analysis of the path selection procedure.** A positive value of the path quality factor indicates that more traffic can be transmitted through the path, without danger of congestion in the network. The negative values are only reached when the load of the path is higher than its transmission capacity (see Eq. 3). From this point of view, if the network is not totally overloaded by prioritised data flows, the quality values of the chosen optimal paths for GC flows should be positive most of the time. Small positive quality values indicate that the variance of the load of the prioritised traffic is low and the control system can share the free capacity optimally. On the other hand, large positive values indicate that the network is heavily underloaded.

**Table 2.** Description of the test results

Test case {Dist. between switches (km), optimal buff. size, packet loss level}	Avg. Throughput	Packet loss ratio of GC traffic	Avg. occupancy of GC buffers of edge switches (normalized to optimal buff. size)	Avg. occupancy of GC buffers of edge switches (normalized to optimal buff. size)
{100,500,1500}	0.959	0.00003	0.07	0.19
{100,1000,3000}	0.969	0	0.10	0.22
{300,500,1500}	0.966	0.00001	0.10	0.23
{300, 1000, 3000}	0.972	0.00001	0.12	0.24

The distribution of the quality values of the chosen paths during one test case is described in Fig. 3. The distribution of the quality values indicates that the path selection procedure have found some underloaded path as an optimal path in most of the control cases. However, the highest probability density is around the quality value zero. This means that in most of the control cases even the optimal paths have been loaded close to capacities of the paths.



**Fig. 3.** The probability density distribution of the quality values of the optimal paths

**4. Conclusions**

In this paper, we have presented the distributed data rate and path selection control system for the generic controllable (GC) traffic in a generic packet switched backbone network. The key idea of the control system is to control both the data rates and paths of the GC traffic, so that the free capacity of the network, after the load of the high priority traffic, can be optimally used for the GC traffic. The control functions are based on the use of fuzzy controllers, ANFIS neuro-fuzzy inference system and statistical calculations. The control functions of the control system are distributed to the edge switches of the network.

According to tests, the mean throughputs of the GC traffic flows are about 0.97 of the theoretical maximum throughput value calculated according to the max-min principle. Although the throughput values are high, the control system can avoid congestion in most of the load situations. However, more research work is needed to study the problems of the dynamic path selection procedure.

## References

1. Paul Ferguson, Geoff Huston. Quality of Service, DeliveringQoS on the Internet and in Corporate Networks. Wiley Computer Publishing, John Wiley & Sons, Inc. 1998, Chapter 9.
2. Ahmet Sekercioglu, Y.; Pitsillides A., Intelligent control techniques for efficient regulation of ABR queue length in ATM switches, Proceedings. Second IEEE Symposium on Computers and Communication (Cat. No. 97TB100137), p 80-4
3. C. Douligeris, G. Develekos. A fuzzy logic approach to congestion control in ATM networks. 1995 IEEE International Conference on Communication, Seattle, pp. 1969-73, Vol. 3.
4. A. Pitsillides, Y. A. Sekercioglu, and G. Ramamurthy. Effective Control of Traffic Flow in ATM Networks Using Fuzzy Explicit Rate Marking (FERM) IEEE Journal on Selected Areas of Communications. Vol. 15, No. 2, February 1997, pp. 209 – 225.
5. G. Ascia, G. Ficili, D. Panno. Design of a VLSI Fuzzy Processor for ATM Traffic Sources Management. Proc. of the 20Th Conference on Local Computer Networks, Minneapolis, Oct. 1995, pp. 62-71
6. Aboelela, E.; Douligeris, C. Fuzzy Metric Approach for Routing in B-ISDN. ICC '99. 1999 IEEE International Conference on communications. 6-10 June, 1999 p. 484 – 488
7. J.-S. R. Jang. ANFIS: Adaptive-Network-Based Fuzzy Inference System. IEEE Transactions on Systems, Man, and Cybernetics. Vol. 23 no. 3. May/June 1993.
8. J.-S.R. Jang, C.-T. Sun, Predicting chaotic time series with fuzzy if-then rules. Second IEEE International Conference on Fuzzy Systems 1993. Pages 1079-1084. vol 2.
9. D. Bertsekas, R. Gallager. Data Networks. Second edition. Prentice-Hall, Englewood Cliffs, New Jersey, 1992. Section 6.5.2.
10. Mischa Schwartz. Broadband Integrated Networks. Prentice-Hall, A Simon & Schuster Company, Upper Saddle River, New Jersey.

# Application-Independent End-to-End Security in Shared-Link Access Networks

José C. Brustoloni and Juan A. Garay

Bell Laboratories – Lucent Technologies  
600 Mountain Avenue  
Murray Hill, NJ 07974, USA  
{jcb,garay}@research.bell-labs.com

**Abstract.** ISPs now offer Internet access via cable modem or DSL, which provide much higher bandwidth than does PSTN. Higher access bandwidths allow ISP customers to exploit NAT (network address and port translation) to amortize the cost of an ISP account among multiple computers. The reduced per-computer cost may encourage airport lounges, hotels, and other businesses that serve “road warriors” to provide Internet connectivity to their clients. Unfortunately, NAT may not interoperate with IPSec, which provides application-independent security in VPNs (virtual private networks). A VPN is necessary, e.g., to connect a “road warrior” securely to a corporate Intranet via the untrusted Internet. We propose a simple DHCP extension that allows client IPSec implementations to interoperate with NAT. The resulting architecture, EASE, makes “road warrior” access easy, secure, and economical.

## 1 Introduction

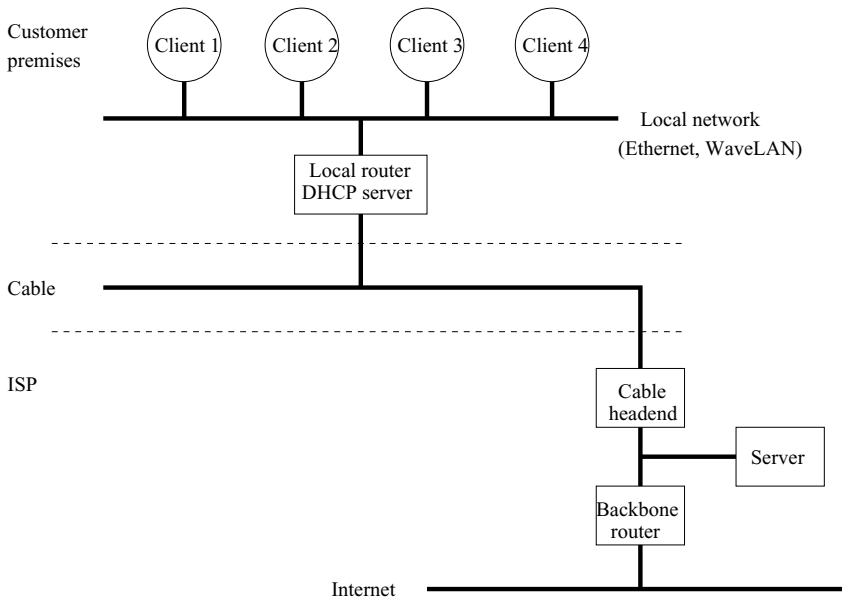
Many ISPs (Internet service providers) still offer Internet access via PSTN (public switched telephone network) lines, which provide low bandwidth (at most 57 Kbps). Recently, however, ISPs began offering Internet access via cable modem or DSL (digital subscriber line). The latter alternatives provide much higher bandwidth (up to several Mbps) at only slightly higher price.<sup>1</sup>

Higher bandwidths make it practical to share a single access link and ISP account among multiple computers. Sharing is implemented by NAT (network address and port translation) [8] and reduces the per-computer Internet connectivity cost. The reduced cost may encourage businesses that serve “road warriors,” such as airport lounges, hotels, and conference centers, to provide Internet connectivity to their clients (“road warriors” are people who need to work away from their offices).

However, NAT is assumed to be incompatible with IPSec (the IP security architecture) [14,15,6] and therefore unsuitable for “road warriors.” IPSec provides

---

<sup>1</sup> For example, in the United States, monthly flat fees in July of 1999 were around \$14 for a PSTN line and \$20 for a PSTN ISP account, versus \$40 for cable service and cable ISP account.



**Fig. 1.** The EASE architecture allows multiple clients to share easily, securely, and economically a single high-bandwidth access link and ISP account.

application-independent security in VPNs (virtual private networks). A VPN is necessary, e.g., to connect a “road warrior” securely to a corporate Intranet via the untrusted Internet.

*Contributions of this paper.* This paper proposes a simple DHCP (dynamic host configuration protocol) [7,1] extension that enables client IPSec implementations to fully interoperate with NAT, making “road warrior” connectivity easy, secure, and economical. The proposed extension is implemented in the EASE architecture, as illustrated in Fig. 1.

EASE uses a *shared-link* access network, where multiple hosts may be dynamically connected to a local network (e.g., Ethernet or WaveLAN). A local router connects the local network to an ISP via a shared high-bandwidth link (e.g., cable, DSL, or T1 line). EASE provides easy connectivity because its local router incorporates a DHCP server, which automatically provides to dynamically connected client hosts the necessary networking configuration (e.g., IP address and default router). The local router also implements NAT, reducing the per-host Internet connectivity cost.

Security is the biggest hurdle in an architecture such as EASE’s. Because client hosts are connected to a local network, an airport lounge that adopts such an architecture might allow, for example, a passenger to forge or snoop on another passenger’s packets. Preventing forgery and snooping requires authentication and encryption, respectively. Clients may use IPSec to obtain the required end-to-end security (authentication and/or encryption) without modifications to



applications. Our proposed DHCP extension makes it possible for IPSec to interoperate with NAT, achieving easy, secure, and economical connectivity.

*Related work.* There are many alternatives to IPSec. For example, SSH (Secure Shell) [17] and SSL (Secure Sockets Layer) [10] implement security in the application layer. SSH provides security for applications such as logging into a remote computer, executing commands in the remote computer, and transferring files between the local and the remote computers. SSL was introduced by Netscape and is widely used for Web applications. Unlike such protocols, IPSec implements security at the network layer and has the advantage of being application-independent.

PPTP (Point-to-Point Tunneling Protocol) [23] was introduced by Microsoft and allows “road warrior” users to connect with corporate Intranets via the Internet. PPTP can make long-distance calls into corporate Intranets unnecessary (a local call into an ISP suffices). PPTP can also spare corporate Intranets the cost of access routers. IPSec can provide similar benefits in tunnel mode. However, IPSec is vendor-independent, seemingly more secure [3], and, unlike PPTP, can also provide end-to-end security (in transport mode, as explained in Section 3).

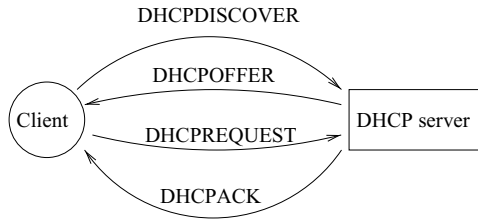
The DHCP extension proposed in this paper can reduce access costs and might allow access services to be provided on a complimentary basis. In another paper [2], we describe how businesses that serve “road warriors” may provide Internet access to their clients and charge for such access.

*Organization of the paper.* The rest of this paper is organized as follows. Sections 2, 3, and 4 discuss in greater detail DHCP, IPSec, and NAT, respectively. Section 5 describes VPN masquerade, a NAT implementation that provides limited interoperability with IPSec. Section 6 describes our proposed DHCP extension, which builds on VPN masquerade to provide full interoperability and backward compatibility. Section 7 presents a summary and final remarks.

## 2 DHCP

This section describes DHCP in greater detail. DHCP is a protocol that allows *client* hosts to obtain configuration parameters from *server* hosts. In the EASE architecture, client hosts use DHCP to obtain and maintain their networking configuration, including client IP address, network address mask, broadcast address, and IP addresses of the router, DNS (domain name system) server, NTP (network time protocol) server, and (possibly) the line printer server assigned to the client. DHCP is what makes EASE easy to use: Clients can, for example, simply connect their laptops to the Ethernet or WaveLAN in an airport lounge or conference room, reboot the computer, and automatically be ready to access the Internet.

DHCP is layered on top of UDP. DHCP clients and servers use UDP ports 68 and 67, respectively. DHCP packets have a format similar to that of BOOTP



**Fig. 2.** Clients automatically obtain their networking configuration from a DHCP server.

(boot protocol) [4]; the same format is used both for client requests and server responses [7].

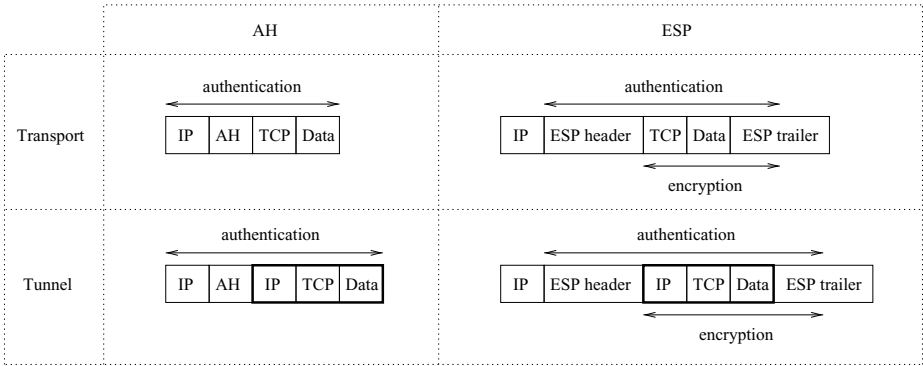
When a DHCP client boots, it broadcasts in the local network a DHCP packet of type DHCPDISCOVER, as shown in Fig. 2. This causes one or more DHCP servers to send to the client a packet of type DHCPOFFER. The client then broadcasts a DHCPREQUEST packet, specifying the selected server in the packet's "server identifier" option. That server then commits the configuration by replying DHCPACK to the client.

If the local network does not contain an active DHCP server, the local router (or another BOOTP or DHCP agent) *relays* DHCP client requests to the appropriate DHCP server. The agent marks its address in the packet's **giaddr** (gateway IP address) field. This allows the actual DHCP server to return the packet to the agent, instead of attempting to reply directly to the client. The agent then returns the reply to the client, using the client's MAC address.

DHCP supports three mechanisms for client IP address allocation: *manual*, *automatic*, or *dynamic*. In manual allocation, each client's IP address is assigned by the network administrator, and DHCP is used only for centralizing such configuration. In automatic allocation, the DHCP server itself selects a permanent IP address for each client. Finally, in dynamic allocation, the DHCP server assigns an IP address to a client only for the period of time specified in the DHCP packet's "IP address lease time" option. To keep its IP address, the client must initiate another DHCPREQUEST before the lease expires. Clients may also explicitly release an IP address by sending to the server a DHCPRELEASE packet. The DHCP server may reuse client IP addresses after they are expired or released. EASE uses DHCP's dynamic allocation.

### 3 IP Security

As mentioned in the previous section, EASE uses DHCP for automatic networking configuration of, for example, client laptops dynamically connected to the Ethernet or WaveLAN in an airport lounge, hotel, or conference room. Because in such applications a client might easily snoop on another client's packets, EASE uses IP Security (IPSec) to provide the necessary security. This section summarizes the IPSec fundamentals.



**Fig. 3.** IPsec packet format depends on protocol (AH or ESP) and mode (transport or tunnel). The portion of the packet that is authenticated or encrypted is different for AH or ESP. The encapsulated packet is shown in bold.

IPsec is an Internet standard from the IETF IPsec Working Group [16]. IPsec is a mandatory part of the next-generation IP protocol (IPv6 [5]), but most existing IPsec implementations assume current-generation IP (IPv4). IPsec operates at the network layer and therefore is independent of the transport- (e.g., TCP or UDP) or application-layer protocol (e.g., HTTP, FTP, or TELNET). IPsec is essentially an *encapsulation* protocol, namely, one that defines the syntax and semantics of placing one packet inside another. IPsec defines two protocols, AH (Authentication Header) [14] and ESP (Encapsulating Security Payload) [15]. AH can provide authentication of packet origin, proof of integrity of packet data, and protection against packet replay. ESP can provide, in addition to AH’s services, encryption of packet data and limited traffic flow confidentiality.

AH and ESP can be used either in *transport* or *tunnel* mode. Transport mode provides end-to-end security between the packet’s source and destination. In contrast, tunnel mode *encapsulates* packets and thus provides security between the nodes where the packet is encapsulated and decapsulated (these can be any nodes, e.g. routers, on the path between the packet’s source and destination). In EASE, a “road warrior” client might use, for example, transport mode to download (via FTP) a document from a supplier’s server. On the other hand, a client would use tunnel mode to connect to an IPsec gateway into the Intranet of the client’s employer.

The packet layout depends on the protocol and mode, as shown in Fig. 3. In IPv4, AH and ESP are identified by values 51 or 50 in the IP header’s protocol field, respectively. AH and ESP insert a header between the IP header and the upper-layer header (in transport mode) or the encapsulated IP datagram (in tunnel mode). ESP also appends a packet trailer. Note that, in tunnel mode, the IP header may have source and destination IP addresses different from those of the encapsulated packet.

AH's authentication covers the whole IP datagram, as illustrated in Fig. 3. In contrast, ESP's authentication skips the IP header and the final part of the ESP trailer (which contains the authentication data). ESP's encryption skips both IP and ESP header and the final part of the ESP trailer.

IPSec peers negotiate what security services to implement (e.g., authentication and/or encryption) and what algorithms and keys to use. In addition to MD5 [19] and SHA [20] for authentication and DES [21] for encryption, IPSec implementations may support other algorithms. The choice of services, algorithms, and keys is called a *security association* (SA). The framework for SA negotiation is defined by ISAKMP (Internet Security Association and Key Management Protocol) [22]. ISAKMP is layered on top of UDP and uses UDP port 500 both for source and destination. IPSec's negotiation is more specifically defined by IKE (Internet Key Exchange) [12]. An IPSec packet's SA is uniquely identified by the protocol (AH or ESP) and destination IP address in the IP header, in conjunction with the SPI (Security Parameters Index, a 32-bit field) in the AH or ESP header.

## 4 NAT

Although DHCP makes EASE's configuration easy and IPSec makes EASE's communication secure, EASE would still be impractical if a separate ISP account were necessary for each EASE client. Because NAT allows EASE clients to share a single ISP account, NAT makes EASE convenient and economical. This section explains how NAT works.

NAT allows local hosts to use each a *private* IP address. Private addresses were reserved by the Internet Assigned Numbers Authority (IANA) for non-exclusive private use [25]. Private addresses spare EASE installations of the burden of obtaining globally unique IP addresses for each client. Dynamically connected clients obtain from EASE's DHCP server locally unique private IP addresses. When EASE clients need to communicate with other hosts on the Internet, they must (at least temporarily) use a *global* IP address (which is globally unique and, therefore, routable). EASE obtains global IP addresses from the ISP.

NAT is implemented in the local router between the local network and the ISP and provides the necessary translations between private and global addresses. NAT uses the upper-layer (e.g. TCP or UDP) port number to distinguish packets of the various local hosts<sup>2</sup>. In *outgoing* traffic (packets sent to the ISP), NAT modifies each packet header's *private* source (IP address, port number) to a *global* source (IP address, port number). NAT maintains in a *translation table* the one-to-one correspondence between private and global (IP address, port

---

<sup>2</sup> When NAT was originally proposed [8], it used a pool of global addresses and thus might not require port translation. For economic reasons, however, it became more usual to use a single global IP address (or a small number of global IP addresses), along with port translation.

number) pairs. When NAT receives corresponding *incoming* traffic (packets received from the ISP), NAT modifies the packet header's destination from global to private (IP address, port number).

Some application-layer protocols, e.g. FTP (File Transfer Protocol) [24], may include in packet payloads IP addresses and possibly port numbers. Such addresses and port numbers must also be translated. Therefore, for each such protocol, NAT includes an Application Level Gateway (ALG) that provides the necessary translations.

Note that DHCP and NAT give to EASE clients a degree of anonymity. In a hotel, for example, a given private IP address could at any time be allocated to any guest, and the hotel's global IP address could be simultaneously used by all guests. This anonymity is usually advantageous.

Most NAT implementations do not support IPSec. In fact, it is widely believed that IPSec cannot interoperate with NAT [6]. The next section shows, however, that under certain conditions, some partial interoperation is possible. In Section 6 we show how to achieve full IPSec functionality with NAT.

## 5 VPN Masquerade

Several difficulties suggest that interoperation of IPSec with NAT is not possible. AH's authentication covers the entire packet, including source and destination IP addresses. When NAT translates an address, it would need to adjust AH's authentication data correspondingly. Unfortunately, that is not possible, because NAT does not (and should not) have access to the authentication key. In contrast, ESP's authentication does not cover the IP header. However, ESP interoperation with NAT can still be problematic in transport mode: When NAT translates the source or destination IP address, it would need to adjust the TCP or UDP checksum correspondingly. (TCP and UDP checksums are calculated over the packet's IP "pseudo-header," TCP or UDP header, and data. The pseudo-header includes the source and destination IP addresses.) However, because the checksum is encrypted (along with the rest of the TCP or UDP header and data) but NAT does not have access to the encryption key, NAT would be unable to make the necessary adjustment. Another problem with both AH and ESP is that, unlike TCP and UDP, they do not use "port numbers" that NAT could modify and use for demultiplexing incoming traffic.

"VPN masquerade" [11] is a patch for Linux that, unlike other NAT implementations, does support IPSec, but only for the case of ESP in tunnel mode (and not ESP in transport mode or AH). NAT is possible in this case because, in tunnel mode, the IP pseudo-header of the encapsulated packet is unaffected by NAT's address translations, and therefore no adjustments are necessary in encapsulated checksums.<sup>3</sup>

VPN masquerade does not attempt to translate TCP or UDP port numbers of encapsulated packets, which may be authenticated and/or encrypted. Instead,

---

<sup>3</sup> In this case, however, the anonymity provided by NAT is lost, as the private IP address is sent unchanged in the encapsulated packet.

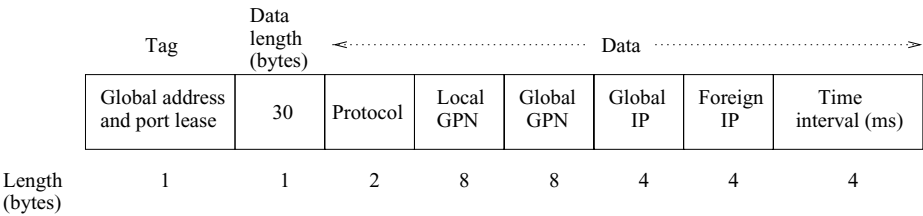
VPN masquerade resorts to a number of heuristics for demultiplexing incoming packets. The heuristics may fail only when two or more local hosts communicate with the same remote node, but even then the probability of failure is low. When the heuristics fail, an incoming packet may be forwarded to the wrong local host. This vulnerability could be used in denial-of-service attacks, but does not compromise integrity or privacy any more than snooping on the local network would.

VPN masquerade treats ISAKMP packets as a special case: UDP packets with source and destination port 500 do not have the port translated. If the packet is outgoing, VPN masquerade writes down in the translation table the local and foreign addresses and the “initiator cookie,” a 64-bit field present in all ISAKMP packets during a negotiation. The local address is private, while the foreign address is global and corresponds to a node outside the local network, with which the local host wishes to communicate. The initiator cookie is randomly selected by the local host. When VPN masquerade receives an incoming ISAKMP packet, it forwards the packet to the local address that corresponds to the packet’s foreign address and initiator cookie. The packet may be incorrectly forwarded if more than one local host is negotiating with the same foreign node using the same initiator cookie.

VPN masquerade uses the foreign address and the SPI field in the ESP header to demultiplex incoming ESP packets. However, VPN masquerade has to determine the corresponding local address by inspection, because the portion of ISAKMP packets that specifies SPI values is encrypted. Additionally, if more than one local host chooses the same incoming SPI for communicating with the same foreign host, VPN masquerade may not demultiplex incoming packets correctly.

Items in VPN masquerade’s translation table associate local address, foreign address, outgoing SPI, and incoming SPI. VPN masquerade marks a translation table item “outstanding” when the first outgoing packet between the given local and foreign addresses and with the given outgoing SPI is forwarded. The incoming SPI is set to 0, as it is then unknown. At most one item with a given foreign address can be outstanding at any time. When the first packet is received from a given foreign address with a given incoming SPI, VPN masquerade forwards the packet to the local address that has an outstanding item with that foreign address. VPN masquerade then updates the item’s incoming SPI and marks the item “established.” If there is no outstanding item with the given foreign address, VPN masquerade multicasts the incoming packet to all local addresses that have recently had an ISAKMP negotiation with the given foreign address. However, because of validation of the cryptographic transformations, the incoming packet will be accepted only by its intended recipient, and dropped by the other multicast recipients. To prevent denial-of-service attacks, VPN masquerade expires outstanding items after a short time, and established items after a period of inactivity.

Unfortunately, VPN masquerade’s scheme is susceptible to race conditions that may cause misassociations. For example, if local hosts *a* and *b* both nego-



**Fig. 4.** Our new DHCP option allows clients to lease global IP addresses, initiator cookies, and incoming SPIs for a specified time interval. Knowledge of the global IP address allows clients to use ESP in transport mode or AH. Cookie and SPI leases prevent NAT demultiplexing errors.

tiate ISAKMP with a foreign node  $f$  at the same time, and  $a$  sends an ESP packet to  $f$  before  $b$  does,  $a$  obtains an outstanding translation table item for  $f$ . However, if  $f$  sends a packet to  $b$  before replying to  $a$ , then  $b$ 's packets will be erroneously forwarded to  $a$  and vice-versa. Misassociations are eventually cleared by timeouts.

## 6 DHCP Extension for IPSec/NAT Interoperation

The previous section describes VPN masquerade, a Linux patch that allows ESP in tunnel mode to interoperate with NAT under certain conditions. We describe in this section a new DHCP extension that builds on that previous work so as to provide full IPSec/NAT interoperability, including AH and ESP protocols both in transport and tunnel mode.

Our DHCP extension consists of a new DHCP option, GLOBAL\_ADDRESS\_AND\_PORT\_LEASE, as illustrated in Fig. 4. DHCP clients use this option to indicate to the DHCP server that, during the specified time interval, they intend to use the specified protocol, global IP address, and private and global GPN (generalized port number) to communicate with the specified foreign IP address. GPN is interpreted according to the protocol: for TCP and UDP, GPN is the TCP or UDP port number; for ISAKMP, GPN is the initiator cookie; for AH and ESP, GPN is the incoming SPI. Clients must specify the protocol and private GPN, but may leave unspecified (that is, with value 0) the time interval, global IP address, and global GPN parameters. The server picks appropriate values for unspecified parameters, and includes them in the reply. If the foreign IP address has value 0, it is a wild card that matches any foreign IP address.

A client sends to its default router the DHCPREQUEST packet with the GLOBAL\_ADDRESS\_AND\_PORT\_LEASE option. The router incorporates a DHCP server and VPN masquerade. The DHCP server processes the option if VPN masquerade is not *nested* (that is, the router does not connect the client's network to another link whose interface has a private IP address). On the other hand, if VPN masquerade is nested (e.g., the ISP provides only private IP ad-

dresses) then the DHCP server *relays* the request to another DHCP server (e.g., located at the ISP's cable head-end or backbone router).

Processing of the GLOBAL\_ADDRESS\_AND\_PORT\_LEASE option is as follows. If the client fully specifies the quadruplet (protocol, global GPN, global IP address, foreign IP address), the DHCP server first checks if the quadruplet is already assigned to the client in VPN masquerade's translation table (i.e., the client is renewing its lease). If so, the DHCP server extends the validity of that translation entry for the requested time interval (picking a default value if unspecified by the client) and returns DHCPACK to the client. Otherwise, the DHCP server checks if the client's total number of items in VPN masquerade's translation table is below a certain limit. If not, the DHCP server returns DHCPNACK to the client. Otherwise, the DHCP server attempts to pick global GPN and/or global IP address (if unspecified by the client) such that the quadruplet (protocol, global GPN, global IP address, foreign IP address) does not conflict with any other current assignment. If the client-specified or server-picked quadruplet has no conflicts, the DHCP server assigns the quadruplet to the client, installs the corresponding new item in VPN masquerade's translation table, and returns DHCPACK to the client. Otherwise, the DHCP server returns DHCPNACK to the client.

In order to interoperate fully with NAT, IPSec implementations should use the new DHCP option when the source IP address is private but the destination IP address is global, or vice-versa. In such cases, NAT is necessary, and IPSec should:

1. Before using an initiator cookie in an ISAKMP negotiation, lease the local host's global IP address and cookie from the DHCP server. This prevents NAT demultiplexing errors due to two or more local hosts using the same global IP address and cookie.
2. For similar reasons, before selecting an incoming SPI in an ISAKMP negotiation, lease the incoming SPI from the DHCP server (keeping the global IP address the same as in the first step).
3. For outgoing packets, before authentication and encryption, (i) in transport mode, replace source port number by a global port number; (ii) in tunnel mode, replace encapsulated source IP address and port number by a global IP address and port number; (iii) sum to the TCP or UDP checksum (a) the difference between global and private source IP addresses, and (b) the difference between global and private source port numbers; and (iv) process any ALG that may be necessary (e.g., for FTP packets).
4. Compute a packet's AH authentication data as if the source or destination IP address (for outgoing or incoming packets, respectively) were equal to the global IP address leased in the first step.
5. For incoming packets, after authentication and decryption, (i) process any ALG that may be necessary (e.g., for FTP packets); (ii) in transport mode, replace global destination port number by the corresponding private port number; (iii) in tunnel mode, replace in decapsulated packet the global destination IP address and port number by the corresponding private address



and port number; and (iv) subtract from the TCP or UDP checksum (a) the difference between global and private destination IP addresses, and (b) the difference between global and private destination port numbers.

6. Periodically renew leases for global IP addresses, initiator cookies, incoming SPIs, and global port numbers, while needed.

Note that TCP and UDP checksum arithmetic uses 16-bit 1-complement arithmetic.

Denial of service attacks are possible because, for example, DHCP packets are not authenticated. However, local and remote hosts or gateways establish cryptographic keys and SAs through ISAKMP, identifying and authenticating each other by means other than IP addresses. Therefore, intruders cannot jeopardize the packet authentication and/or privacy provided by IPSec.

Only minimal modifications are necessary to VPN masquerade. When the DHCP server installs a new AH or ESP item in VPN masquerade's translation table, the DHCP server marks the item "established" and sets its outgoing SPI to 0, a wild card that matches any outgoing SPI for the given local and foreign addresses (thus circumventing VPN masquerade's "outstanding" marking). VPN masquerade should demultiplex AH and ESP packets according to the foreign address and incoming SPI, and translate between private and global IP addresses according to the translation table.

Clients that adopt the new DHCP option can use the previously unsupported AH protocol and/or transport mode and prevent the race conditions and demultiplexing errors discussed in the previous section; also note that the anonymity provided by NAT is now preserved. However, our solution is backward-compatible and continues to support clients that use ESP in tunnel mode and that are not updated to take advantage of the new DHCP option.

## 7 Conclusions

EASE is an architecture that can greatly reduce the cost of Internet access and may allow airport lounges, hotels, and conference centers to provide convenient Internet connectivity. The technology for ubiquitous Internet access is largely available: Cable and DSL provide high-bandwidth, low-cost links to the Internet; NAT allows those links to be shared; DHCP provides automatic configuration; WaveLAN connects clients without wires; and IPSec makes it all secure, regardless of the application. EASE's biggest hurdle is the interoperation of IPSec and NAT. We proposed a simple, backward-compatible DHCP extension that provides full IPSec/NAT interoperation, including the AH and ESP protocols both in transport and in tunnel mode.

## Acknowledgements

The authors thank Dale Blodgett, David Faucher and Dan Heer for their valuable input and many discussions on IPSec and NAT.

## References

1. S. Alexander and R. Droms. "DHCP Options and BOOTP Vendor Extensions," IETF, RFC 2132, Mar. 1997.
2. J. Brustoloni and J. Garay. " $\mu$ ISPs: Providing Convenient and Low-Cost High-Bandwidth Internet Access," to appear in *Proc. 9th Intl. World Wide Web Conf.*, W3C, Amsterdam, Netherlands, May 2000.
3. Counterpane. "PPTP Crack," available at <http://www.counterpane.com/pptp.html>.
4. W. Croft and J. Gilmore. "Bootstrap Protocol," IETF, RFC 951, Sept. 1985.
5. S. Deering and R. Hinden. "Internet Protocol, Version 6 (IPv6) Specification," IETF, RFC 2460, Dec. 1998.
6. N. Doraswamy and D. Harkins. "IPSec: The New Security Standard for the Internet, Intranets and Virtual Private Networks," Prentice-Hall, 1st. ed., July 1999.
7. R. Droms. "Dynamic Host Configuration Protocol," IETF, RFC 2131, Mar. 1997.
8. K. Egevang and P. Francis. "The IP Network Address Translator (NAT)," IETF, RFC 1631, May 1994.
9. FreeS/WAN. Homepage at <http://www.xs4all.nl/~freeswan/>.
10. A. Freier, P. Karlton and P. Kocher. "The SSL Protocol Version 3.0," Netscape, Mar. 1996, available at <http://home.netscape.com/eng/ssl3/ssl-toc.html>.
11. J. Hardin. "Linux VPN Masquerade." Homepage at [http://www.wolfenet.com/~jhardin/ip\\_masq-vpn.html](http://www.wolfenet.com/~jhardin/ip_masq-vpn.html).
12. D. Harkins and D. Carrel. "The Internet Key Exchange (IKE)," IETF, RFC 2409, Nov. 1998.
13. Internet Software Consortium. Homepage at <http://www.isc.org/>.
14. S. Kent and R. Atkinson. "IP Authentication Header," IETF, RFC 2402, Nov. 1998.
15. S. Kent and R. Atkinson. "IP Encapsulating Security Payload (ESP)," IETF, RFC 2406, Nov. 1998.
16. S. Kent and R. Atkinson. "Security Architecture for the Internet Protocol," IETF, RFC 1825, March 1997.
17. T. König. "Ssh (Secure Shell) FAQ - Frequently asked questions," available at <http://www.uni-karlsruhe.de/~ig25/ssh-faq/>.
18. Lucent InterNetworking Systems. Homepage at <http://www.lucent.com/dns/products/>.
19. C. Madson and R. Glenn. "The Use of HMAC-MD5-96 within ESP and AH," IETF, RFC 2403, Nov. 1998.
20. C. Madson and R. Glenn. "The Use of HMAC-SHA-1-96 within ESP and AH," IETF, RFC 2404, Nov. 1998.
21. C. Madson and N. Doraswamy. "The ESP DES-CBC Cipher Algorithm with Explicit IV," IETF, RFC 2405, Nov. 1998.
22. D. Maughan, M. Schertler, M. Schneider and J. Turner. "Internet Security Association and Key Management Protocol (ISAKMP)," IETF, RFC 2408, Nov. 1998.
23. Microsoft. "Point-to-Point Tunneling Protocol (PPTP) FAQ," available at <http://www.microsoft.com/NTServer/commserv/deployment/moreinfo/PPTPfaq.asp>.
24. J. Postel and J. Reynolds. "File Transfer Protocol," IETF, RFC 959, Oct. 1985.
25. Y. Rekhter, B. Moskowitz, D. Karrenberg, G. J. de Groot and E. Lear. "Address Allocation for Private Internets," IETF, RFC 1918, Feb. 1996.

# Fair Efficient Call Admission Control Policies for Heterogeneous Traffic Streams in a Packet Routing Server Using The DTM Technology

Chih-Jen Chang<sup>1</sup> and Arne A. Nilsson<sup>2</sup>

<sup>1</sup> Network Research Laboratory, Corporate Research Laboratories, Motorola Inc.,  
20 Cabot Blvd., MS: M4-15, Mansfield ,MA 02048, USA  
Chih-jen.Chang@motorola.com

<sup>2</sup> Center for Advanced Computing and Communication, Department of Electrical &  
Computer Engineering, North Carolina State University,  
Raleigh, NC 27695, USA  
Nilsson@ncsu.edu

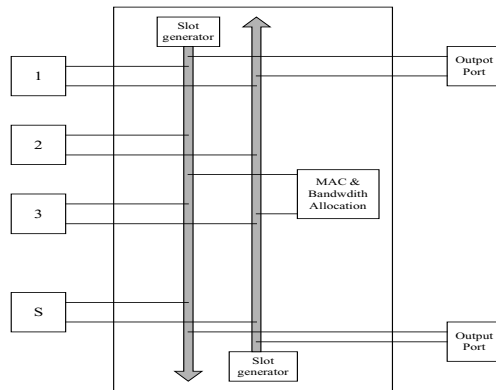
**Abstract.** The issue of fairness has new prominence in multi-service networks where the diverse service characteristics can result in a very unfair resource allocation unless the issue is considered explicitly at the design stage. In this paper, we study the fairness issue of a Packet Routing Server (PRS) architecture based on the Dynamic synchronous Transfer Mode (DTM) access technology. DTM provides a dynamic bandwidth allocation scheme to support various types of telecommunications applications. However, this flexibility also promotes within DTM a lack of fairness concerning medium utilization as no enforcement scheme has been implemented in the DTM access technology. We demonstrate the unfairness of the DTM access scheme and provide resource reservation schemes to ensure that each type of traffic has its relatively fair share of the medium. Our results show that the dynamic resource reservation achieves a better medium utilization and a fair access of the resource in the DTM PRS.

## 1 Introduction

As new and emerging multimedia telecommunication applications and other internet initiated activities have been developed and deployed over the past few years, the services provided by the traditional communication networks have also been expanded into a new dimension. First of all, the expansion of the network services makes the demands on bandwidth become less homogeneous. Also, some new multimedia applications such as voice and video may require special handling in order to maintain the quality of communications. Therefore, communication networks which provide integrated services, support for real-time applications, high speed access, and high bandwidths are needed in order to support the variety bandwidth demands of different applications. One such network based on the Dynamic synchronous Transfer Mode (DTM) [1], [2] access technology has been proposed.

DTM is a shared medium broadband technology based on fast circuit switching [3] and dynamic TDM [4]. Because of its circuit switching nature, it guarantees each host a certain bandwidth or even uses a large fraction of the available bandwidth for effective data transmission. In addition, it provides support for the dynamic allocation of bandwidth such that the networks can adapt to variations in the traffic, and it divide its bandwidth between the hosts according to their demands. The PRS architecture we study in this paper is essentially a shared medium switching mechanism using the DTM access technology as shown in figure 1. The functionality of the PRS can be used as a bridge which connects the different Local Area Networks (LANs) or a router which connect LANs with the WANs [5]. Both implementations are to provide a new way of high speed access for internetworking. An example of the implementation of this PRS is shown in figure 2.

For those networks providing integrated services, the issue of fairness has received some attention in both circuit switched and packet switched networks. However, it has usually been considered a secondary issue as the primary concerns are often to maximize the network throughput in circuit switched networks and to minimize the average delay in packet switched networks. DTM is no exception to this rule, as it provides a mechanism for dynamic bandwidth allocation, but no mechanism to ensure the fair access to the medium. As in the integrated service environment, an ill-behaved user can easily occupy most of the resource if no enforcement has been implemented in the network.

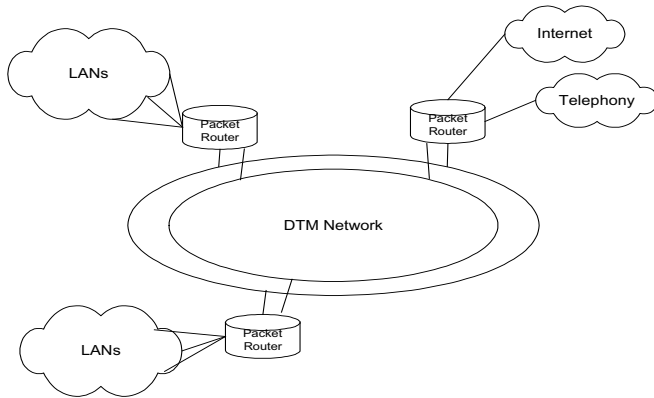


**Fig. 1.** A DTM Packet Routing Architecture

In this paper, we study the fairness issue for the call admission control schemes of a DTM PRS. The fairness issue is based on the utilization of the shared medium as a function of heterogeneous traffic streams. All of the performance comparisons are on the basis of the differing traffic types. We assume that the DTM PRS is basically a multi-service loss system, that is, a host which cannot successfully allocate sufficient bandwidth for data transfer will attempt to allocate the same amount of bandwidth after a random delay. The bandwidth we mentioned above is actually a corresponding

set of 64-bit time slots in a DTM frame (where each frame has a transmission time of  $125 \mu S$ ) and we assume a finite population of traffic sources.

This paper is organized as follows. In Section 2, we recall some basic concepts of call admission control policies which have been used in circuit switching networks and the analytical models for these call admission control policies under the finite population assumption. In Section 3, we demonstrate the unfairness issue in a DTM PRS and provide a solution to provide a fair access of the medium for this DTM PRS. In the final section, we provide a summary of the results in this paper.



**Fig. 2.** Principal view of packet router solution for campus networks

## 2 Call Admission Control Policies

Typically, call admission control policies for circuit switched networks are based upon the derivation of the peak bandwidth of a given source. These policies can be divided into two categories: complete sharing and trunk reservation. In our study, we investigate the fairness issue for three different call admission control schemes originating from these two categories. We describe these schemes in the following sections.

### 2.1 Complete Sharing Policy

We consider a DTM PRS with a fixed bus capacity of  $C$  Mbps. A number  $R$  different traffic classes share the bus in a complete sharing mode as DTM is based on a fast circuit switching scheme and a channel is established for each call. Each type of traffic requires  $b_r$  time slots as  $r=1, \dots, R$ . There are finite number of traffic sources  $S_r$ , where  $r$  varies from 1 to  $R$  for each class  $r$ . The arrival traffic of service class  $r$  is assumed to follow an on-off process with the mean arrival rate  $\beta_r$  per idle source, and mean holding time  $1/\mu_r$ . It is assumed that both the idle time and the busy time

of the on-off process have generally distributions. We let  $C_f$  be the available capacity of the bus upon arrival of a burst of type  $r$ . The request will be blocked if  $C_f < b_r$ ; otherwise, the request is granted. We will refer to this mode as call admission control without reservation.

The analytical model of this call admission control policy is based upon the *Multi-rate Engset* model [6], and it is known to have the feature of being insensitive to the inter-arrival and holding time distribution of the traffic sources [7]. Essentially, the steady state probabilities of this model have a product form solution [8]

$$p(n_1, n_2, \dots, n_R) = p(0) \prod_{r=1}^R \binom{S_r}{n_r} \alpha_r^{n_r} \quad (1)$$

where  $n_r$  is the number of busy sources of class  $r$  and  $\alpha_r$  is defined as  $\alpha_r = \frac{\beta_r}{\mu_r}$ , the offered traffic per idle source of class  $r$ . Following the development presented by Kaufman [9] and Roberts [10], and expanding the dimensionality, a function is introduced in [8] which is defined as follows:

$$q(\underline{S}, m) = \sum_{\sum n_r = m} \prod_{r=1}^R \binom{S_r}{n_r} \alpha_r^{n_r} \quad (2)$$

, where  $\underline{S} = (S_1, S_2, \dots, S_R)$ . The variable  $m$  has the obvious meaning of being the number of busy resources. The blocking probabilities for class  $r$  ( $B_r$ ) are actually the steady state probabilities when an arrival from class  $r$  finds that  $C-m < b_r$ . Therefore,

$$B_r = \frac{\sum_{j=C-b_r+1}^C q(\underline{S}, j)}{\sum_{i=0}^C q(\underline{S}, i)} \quad (3)$$

As indicated in [8], there is a numerical instability with this formula. Therefore, in [8], yet another probabilistic interpretation is also introduced by defining a function as indicated in (4). This function represents the probability of having  $m-k$  servers busy in a system with  $m$  servers. We note that  $0 \leq k \leq m$  and  $\beta(\underline{S}, m, k) = 0$  otherwise.

$$\beta(\underline{S}, m, k) = \frac{q(\underline{S}, m-k)}{\sum_{i=0}^m q(\underline{S}, i)} \quad (4)$$

Recursive equations for the function  $\beta(\underline{S}, m, k)$  were also developed in [8] by using binomial identities as follows:

$$q(\underline{S}, m) = \sum_{i_R=0}^1 \dots \sum_{i_1=0}^1 \left( \prod_{r=1}^R \alpha_r^{i_r} \right) q(\underline{S} - \underline{1}, m - \sum_{r=1}^R i_r b_r) \quad (5)$$

In this paper, we simplify the derivation from that of the original paper. First, we rewrite the  $\beta$ -function as follows:

$$\beta(\underline{S}, m, k) = \frac{q(\underline{S}, m - k)}{\sum_{i=0}^{m-1} q(\underline{S}, i) + q(\underline{S}, m)} \quad (6)$$

Following the binomial identities, the first term in the denominator becomes:

$$\sum_{i=0}^{m-1} q(\underline{S}, i) = \sum_{i=0}^{m-1} \sum_{i_1=0}^1 \sum_{i_2=0}^1 \dots \sum_{i_R=0}^1 \left( \prod_{r=1}^R \alpha_r^{i_r} \right) q(\underline{S} - \underline{1}, i - \sum_{r=1}^R i_r b_r) \quad (7)$$

, and the second term in the denominator is then

$$q(\underline{S}, m) = \frac{1}{m} \sum_{r=1}^R S_r \alpha_r b_r \sum_{\substack{i_l=0, l \neq r \\ l=1, 2, \dots, R}}^1 \left( \prod_{\substack{j=1 \\ j \neq r}}^R \alpha_j^{i_j} \right) q(\underline{S} - \underline{1}, m - b_r - \sum_{\substack{j=1 \\ j \neq r}}^R i_j b_j) \quad (8)$$

and the numerator is:

$$q(\underline{S}, m - k) = \sum_{i_1=0}^1 \sum_{i_2=0}^1 \dots \sum_{i_R=0}^1 \left( \prod_{r=1}^R \alpha_r^{i_r} \right) q(\underline{S} - \underline{1}, m - 1 - (k + \sum_{r=1}^R i_r b_r - 1)) \quad (9)$$

Divided both the numerator and denominator by  $\sum_{i=0}^{m-1} q(\underline{S} - \underline{1}, i)$ , and let

$\beta(\underline{S}, m, k) = \frac{\text{Numerator}}{\text{Denominator}}$ . Thus, the **Numerator** takes on the form:

$$\sum_{i_1=0}^1 \sum_{i_2=0}^1 \dots \sum_{i_R=0}^1 \left( \prod_{r=1}^R \alpha_r^{i_r} \right) \beta(\underline{S} - \underline{1}, m - 1, k + \sum_{r=1}^R i_r b_r - 1) \quad (10)$$

, and the **Denominator** is:

$$\begin{aligned} & \sum_{i=0}^{m-1} \sum_{i_1=0}^1 \sum_{i_2=0}^1 \dots \sum_{i_R=0}^1 \left( \prod_{r=1}^R \alpha_r^{i_r} \right) \beta(\underline{S} - \underline{1}, m - 1, i - \sum_{r=1}^R i_r b_r) + \\ & \frac{1}{m} \sum_{r=1}^R S_r \alpha_r b_r \sum_{\substack{i_l=0, l \neq r \\ l=1, 2, \dots, R}}^1 \left( \prod_{\substack{j=1 \\ j \neq r}}^R \alpha_j^{i_j} \right) \beta(\underline{S} - \underline{1}, m - 1, b_r - 1 + \sum_{\substack{j=1 \\ j \neq r}}^R i_j b_j) \end{aligned} \quad (11)$$

The blocking probabilities can now be computed by using the following equation:

$$B_r = \sum_{k=0}^{b_r-1} \beta(\underline{S}, C, k) \quad (12)$$

## 2.2 Restricted Resource Reservation Policy

Normally a call with higher bandwidth requirement will be blocked with larger probability. To maintain the grade of service for individual classes in a system with heterogeneous traffic streams, a resource reservation mechanism is often used. In this section, we consider the same system as in the previous section with a reservation scheme. In the reservation mechanism, we denote the reservation threshold assigned to class  $r$  by  $T_r$ ,  $r=1,2,\dots,R$ . For those classes which are able to use the reserved resources, no threshold is assigned. We assume that the set of those classes with threshold assigned is denoted as  $A$ . The algorithm works as illustrated in figure 3.

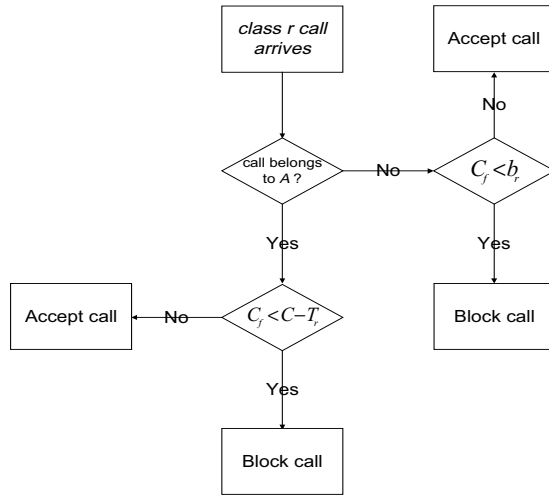


Fig. 3. Restrict Resource Reservation Scheme

As discussed in [11], the general rule for balancing the blocking probabilities of calls from different classes  $i, j, k, \dots$  is to set the corresponding thresholds  $T_i, T_j, T_k, \dots = C - \max\{b_i, b_j, b_k, \dots\}$ . This rule will equalize the blocking probabilities of traffic types  $i, j, k, \dots$  in the case of infinite population.

In the case of trunk reservation in systems with finite population, we consider the Markov chains where each state represents the number of total busy time slots. Because of the reservation mechanism, the states from  $(C - b_R + 1)$  to  $C$  in the Markov



chains do not have transitions to the states within this subset, making the Markov chains irreversible. Hence, product form solutions is no longer valid in this case. However, in principle, we can still obtain a numerical approximation by using the recursive algorithm we introduced in the previous section. Basically, the analysis is to follow the  $\beta(\underline{S}, C, k)$  recursion, however, some modifications need to be made in order to obtain the numerical approximation.

We let  $\beta(\underline{S}, m, k) = \frac{\text{Numerator}}{\text{Denominator}}$ , the modification is:

$$\text{Numerator} = \sum_{i_1=0}^1 \sum_{i_2=0}^1 \dots \sum_{i_R=0}^1 \left( \prod_{r=1}^R \alpha_r^{i_r} \right) \beta(\underline{S} - \underline{1}, m - 1, k + \sum_{r=1}^R i_r b_r - 1) \quad (13)$$

, and the **Denominator** will have the form in equation (15). We note that:

$$b_r(m-k) = \begin{cases} b_r : m-k \leq T + b_r \\ 0 : m-k > T + b_r \end{cases}, r=1, 2, \dots, R \text{ and } T = C - \max(b_1, b_2, \dots, b_R).$$

$$\begin{aligned} & \sum_{i=0}^{m-1} \sum_{i_1=0}^1 \sum_{i_2=0}^1 \dots \sum_{i_R=0}^1 \left( \prod_{r=1}^R \alpha_r^{i_r} \right) \beta(\underline{S} - \underline{1}, m - 1, i - \sum_{r=1}^R i_r b_r) + \\ & \frac{1}{m} \sum_{r=1}^R S_r \alpha_r b_r (m-k) \sum_{\substack{i_j=0, l \neq r \\ l=1, 2, \dots, R}}^1 \left( \prod_{\substack{j=1 \\ j \neq r}}^R \alpha_j^{i_j} \right) \beta(\underline{S} - \underline{1}, m - 1, b_r - 1 + \sum_{\substack{j=1 \\ j \neq r}}^R i_j b_j) \end{aligned} \quad (14)$$

Again, the blocking probabilities for class  $i$  is:

$$B_r = \sum_{k=0}^{\max\{b_r-1, T-1\}} \beta(\underline{S}, C, k) \quad (15)$$

### 2.3 Dynamic Resource Reservation Policy

Although the restricted resource reservation policy can balance the blocking probabilities for different traffic classes, the total medium utilization decreases. However, our goal is not only to provide a technique for fair access of the medium but also to maintain a better medium utilization. Therefore, we introduce the dynamic reservation policy in this section. This mechanism relies on the number of free slots upon the arrival of a call from class  $r$ . We assign the threshold of the system the same as the restricted resource reservation as we wanted to balance the blocking probabilities for different traffic classes. We consider a PRS with  $R$  traffic classes and each require  $b_r$  time slots for transmission. It is also assumed that the number of required time slots for each class have the following relationship:  $b_1 \leq b_2 \leq \dots \leq b_R$ . Thus, we assign the threshold  $T = C - \max\{b_1, \dots, b_R\}$ . Obviously,  $T$  equals to  $C - b_R$  in this case. The call admission algorithm works as illustrated in figure 4. As shown in figure 4, it may

seem that the system still admits calls with lower bit rates when the threshold has been reached or exceeded, thus introducing unfairness to classes with higher bit rates. The argument is that since the number of reserved time slots are not always equal to  $b_R$ , in the case of  $C_f < b_R$ , the resources are wasted and the unfairness still exists under restrict reservation scheme in a system with finite population. The mechanism shown in figure 4 admits calls from a certain class strictly based on the current available resources when the threshold has been reached or exceeded, hence providing higher utilization and better fairness in a systems with finite population. We will show the simulation results of using this reservation mechanism in next section.

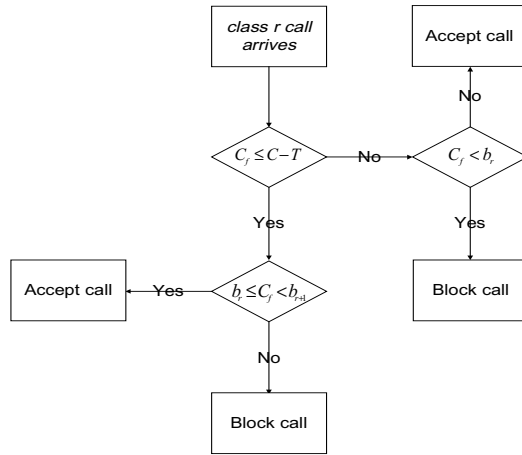


Fig. 4. Dynamic Resource Reservation Scheme.

We note that due to the complexity of this algorithm, only simulation results are shown in this paper.

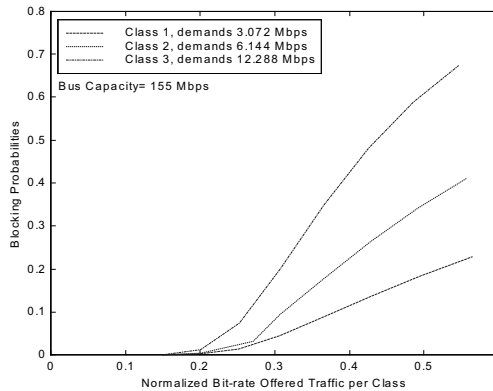
## 2 Numerical Results

First of all, we would like to point out that our previous work [12], [13] has shown that *Multi-rate Engset* model is a valid model for the DTM PRS architecture. Therefore, we only present the analytical results for the call admission control policy without reservation in this paper. We demonstrated the unfairness of medium access in a DTM PRS. The system has two dual fiber bus, each with the capacity of 155 Mbps. We increase the number of traffic sources attached to the PRS thus increasing the offered load to the fiber bus. The offered load is the normalized bit rate traffic for each class and is computed by using:  $\text{Offered Load} = \frac{S_r \alpha_r}{1 + \alpha_r (1 - B_r)} * \frac{b_r}{C}$ , where  $S_r$

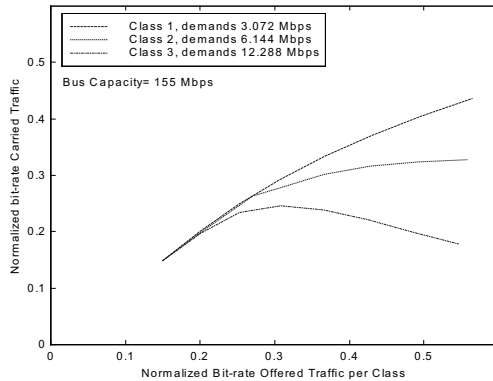
represent the number of traffic sources from class  $r$ , and  $\alpha_r, b_r, B_r$  are the same as defined in section 2.1.  $C$  is the capacity of the fiber bus.

We consider a system with three traffic classes as shown in figures 5 and 6 and assume  $\alpha_1 = 1$ ,  $\alpha_2 = 1/3$  and  $\alpha_3 = 1/7$ . It is also assumed that class 1 requires bandwidth of 3.072 Mbps, class 2 requires bandwidth of 6.144 Mbps and class 3 requires bandwidth of 12.288 Mbps.

Both figures 5 and 6 show that the traffic classes with higher bit rates suffers from higher blocking probabilities and lower throughput at the same value of offered traffic. These indicate the unfairness in the complete sharing scheme.



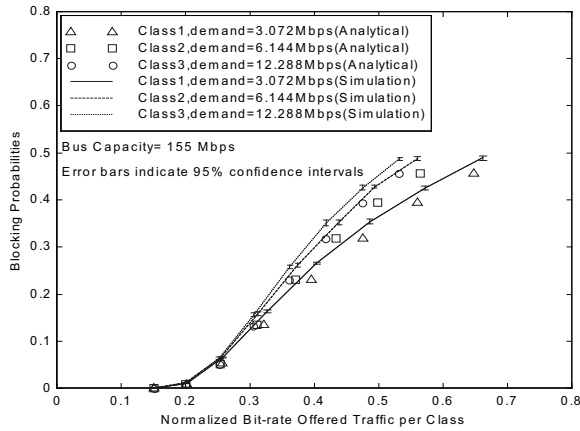
**Fig. 5.** Blocking probabilities for three traffic classes in a DTM PRS without resource reservation.



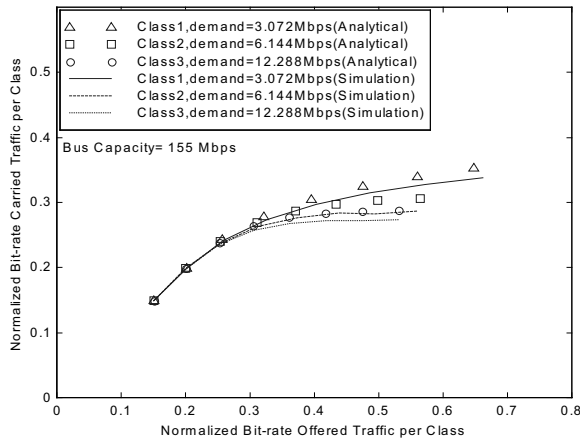
**Fig. 6.** Normalized carried traffic for three traffic classes in a DTM PRS without resource reservation.

We then consider the same system configuration with restricted resource reservation scheme and show the results in figures 7 and 8. As shown in figure 7, the approximation of the blocking probabilities obtained from the recursive formula are close to the simulation results in this system. We also show in figures 7 that the blocking probabilities for different traffic classes are brought closer by employing the

resource reservation policy. However, as indicated in figures 7 and 8, under higher load, this reservation mechanism still suffers from unfairness problem.

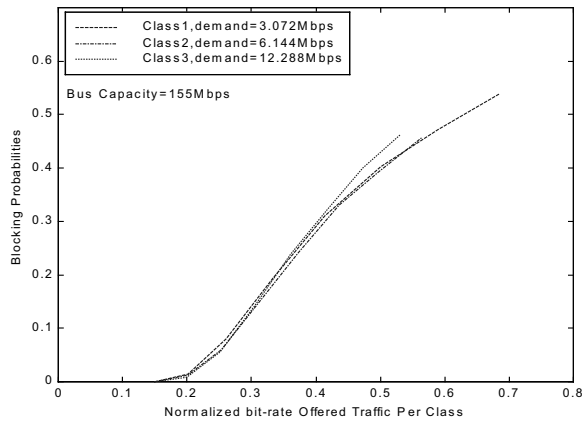


**Fig. 7.** Blocking probabilities for three traffic classes in a DTM PRS with restricted resource reservation.

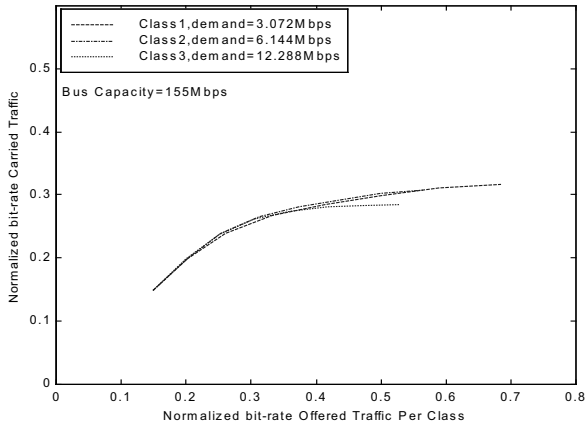


**Fig. 8.** Normalized carried traffic for three traffic classes in a DTM PRS with restricted resource reservation.

Finally, we consider the same system again, however, with dynamic resource reservation scheme. We show the results in figures 9 and 10. Figure 9 shows that the curves of blocking probabilities for different traffic classes are closer with the dynamic reservation mechanism than with the restricted reservation scheme. We also present in figure 10 that the throughputs for different classes are closer under the dynamic reservation technique. Thus, we have achieved a fair medium access and better utilization for heterogeneous traffic streams by employing the dynamic reservation mechanism into the DTM PRS architecture.



**Fig. 9.** Blocking probabilities for three traffic classes in a DTM PRS with dynamic resource reservation.



**Fig. 10.** Normalized carried traffic for three traffic classes in a DTM PRS with dynamic resource reservation.

### 3 Conclusion

In this paper, we have considered a DTM PRS architecture with different types of services. We investigate the fairness issue in this system by employing different call admission control policies including complete sharing, restrict reservation and dynamic reservation. The analytical models and computation schemes addressing the related call admission control policies have been reviewed except for the dynamic reservation due to its complexity. Simulation results have been carried out to illustrate the efficiency and grade of service management capability of resource reservation

mechanisms. In conclusion, the dynamic reservation policy provides a better fairness and medium utilization for the DTM PRS system than the other call admission control policies.

## References

1. Lars H. Ramfelt, "Architecture for High Performance Optical Networks", Doctor of Technology Dissertation, Department of Teleinformatics, The Royal Institute of Technology, Sweden, February 1996.
2. Christer Bohm, "The DTM Protocol - Design and Implementation", Licentiate Thesis, Department of Teleinformatics, The Royal Institute of Technology, Sweden, February 1994.
3. Per Lindgren, "A Multi-Channel Network Architecture Based on Fast Circuit Switching", Doctor of Technology Dissertation, Department of Teleinformatics, The Royal Institute of Technology, Sweden, May 1996.
4. L.T. Wu, S.H. Lee and T.T. Lee, "Dynamic TDM- A Packet Approach to Broadband Networking", *Proceedings of IEEE Globalcom*, Page 1585-1592, 1987.
5. Markus Hidell, "Supporting the Internet Protocols in a High Performance Real-Time Network", *Licentiate Thesis, KTH, Stockholm*, April, 1996.
6. J.W. Cohen, "The generalized Engset formulae", *Philips Telecomm.*, 18 (1957), pp 150-170.
7. Ryszard Syski, "Introduction to Congestion Theory in Telephone Systems", Elsevier Science, 1986.
8. Mark J. Perry and Arne A. Nilsson, Multirate Blocking Probabilities: "Numerically Stable Computations", *ITC15*, pp. 1359 - 1368, June 1997.
9. J.S. Kaufman, "Blocking in a Shared Resource Environment", *IEEE Trans. On Communication*, Vol. 29, October 1981, pp. 1474 -1481
10. J.W. Roberts, "Teletraffic Models for the Telecom 1 Integrated Services Network", *ITC 10*, Montreal 1983, paper 1.1.2.
11. P. Tran-Gia and F. Hübner, "An Analysis of Trunk Reservation and Grade of Service Balancing Mechanisms in Multiservice Broad Networks", *IFIP Transactions C-15*, 1993, pp. 83- 97.
12. C.J. Chang and A.A. Nilsson, "Performance Evaluation of DTM Access Nodes", TR98/05, Center for Advanced Computing and Communications, North Carolina State University, April 1998
13. C.J. Chang and A.A. Nilsson, "Analytical Model for DTM Access Nodes", TR98/11, Center for Advanced Computing and Communications, North Carolina State University, April 1998

# An Agent-Based Framework for Large Scale Internet Applications

Mamadou Tadiou Kone<sup>1</sup> and Tatsuo Nakajima<sup>2</sup>

<sup>1</sup> Japan Advanced Institute of Science and Technology, 1-1 Asahidai,  
Tatsunokuchi-machi, Nomi-gun, Ishikawa-ken, Japan 923-12

`mamadou@jaist.ac.jp`

<sup>2</sup> Waseda University, Shinjuku-ku, Ookubo, Tokyo 169-8555, Japan.

`tatsuo@mn.waseda.ac.jp`

**Abstract.** The idea of a software entity that performs tasks on behalf of a user across the Internet is now well established. We introduce in this paper a new approach to service discovery and QoS negotiation over the Internet. Our approach presents a framework for service discovery and QoS negotiation at the network level that rely on two concepts: multi-agent systems and agent communication languages (ACL). In this framework, a *user* and *service agents* engage in a structured communication through the mediation of a *QoS Broker Agent* and a *Facilitator Agent*. Here, the Facilitator Agent acts on behalf of several service agents. It acquires information from these service agents and acts as a single point of contact to supply this information to the User Agent via the QoS Broker Agent. A number of service discovery protocols like the Service Location Protocol (SLP), and Sun Microsystem's Jini has been designed for restricted environments and do not scale to the entire Internet. In order to provide an infrastructure for large scale Internet applications, we designed a prototype multi-agent system that is able to discover resources and negotiate QoS at the network level.

**Keywords:** Mutli-agent systems, Agent Communication Languages (ACL), Knowledge Query and Manipulation Language (KQML), Quality of Service (QoS), Internet.

## 1 Introduction

The tremendous growth of the Internet in the past few years sparked a whole new range of applications and services based on its technologies. Users will be able to take full advantage of these new capabilities only if there is an appropriate configuration to deal with the scalability and heterogeneity problems inherent to the Internet. In this line, resource discovery on the network and Quality of Service (QoS) assurance are important subjects that are drawing attention. In particular, the Service Location Protocol (SLP) [4] designed by the Internet Engineering Task Force (IETF) aims to enable network-based applications to automatically discover the location of services they need. However, SLP was designed for use in networks where the Dynamic Host Configuration Protocol (DHCP) [1] is available or multicast is supported at the network layer. Neither

DHCP nor multicasting extend to the entire Internet because these protocols must be administered and configured. As a result, SLP does not scale to the Internet.

Our objective in this paper is to deal with two important limitations in resource management for large scale applications: scalability and communication costs. We propose in this paper a framework that relies on the concepts of Multi-agent Systems and Agent Communication Language (ACL) (here the Knowledge Query and Manipulation Language (KQML) described in [3]). In this framework, a user agent, a QoS manager agent, one or several facilitator agents, and service agents (application agent, system agent, network agent, and resource agent) engage in a mediated communication through the exchange of structured KQML messages.

Following this introduction, we state in section 2 the problem we intend to examine. In section 3, we describe the concepts and protocols underlying our multi-agent system based QoS negotiation scheme. In addition, we give the implementation details of our framework in the same section. Some issues and perspectives are proposed in section 4 and then related works are presented in section 5. Finally, we conclude in the last section 6.

## 2 Agent-Based Systems

### 2.1 Multi-agent Systems

There are two well-known perspectives in defining the word *agent*: the software engineering perspective and the cognitive science (AI) perspective. The first refers to a piece of software called *mobile agent* or *autonomous agent* that can migrate autonomously inside a network and accomplish tasks on behalf of their owners. On the other hand, the second states that *multi-agent systems* are distributed computing systems composed of several interacting computational entities called agents. These constituent agents have capabilities, provide services, can perceive and act on their environment. Service components involved in a QoS provision are modeled as this type of agent.

### 2.2 Agent Communication Languages

An agent communication language (ACL) stems from the need for better problem solving paradigms in distributed computing environments. One of the main objectives of ACL design is to model a suitable framework that allow heterogeneous agents to interact, communicate with meaningful statements that convey information about their environment or knowledge.

The Knowledge Sharing Effort group designed one example of ACL, the Knowledge Query and Manipulation Language (KQML) described in [3]. Our framework uses the KQML language, which is made of three layers (figure 1) : the communication layer, the message layer, and the content layer. A KQML message has the following structure:



Communication Layer: sender, receiver, msg id
Message Layer: performatives, msg format
Content Layer: ontology, content language

Fig. 1. KQML three layers structure

(*tell*  
:sender QoS-manager  
:receiver User  
:language Prolog  
:in-reply-to id1  
:ontology QoS-ontology  
:content “available(resource,URL)”  
)

Here, *tell* is called a *performative*, *:sender*, *:receiver*, *:language*, *:in-reply-to*, and *:ontology* are parameters. The QoS manager informs (*tell*) the user about the availability of a resource on the Internet by using Prolog as a content language. There are two types of agent communication: the direct communication relates a sender agent with a known receiving agent and the mediated communication

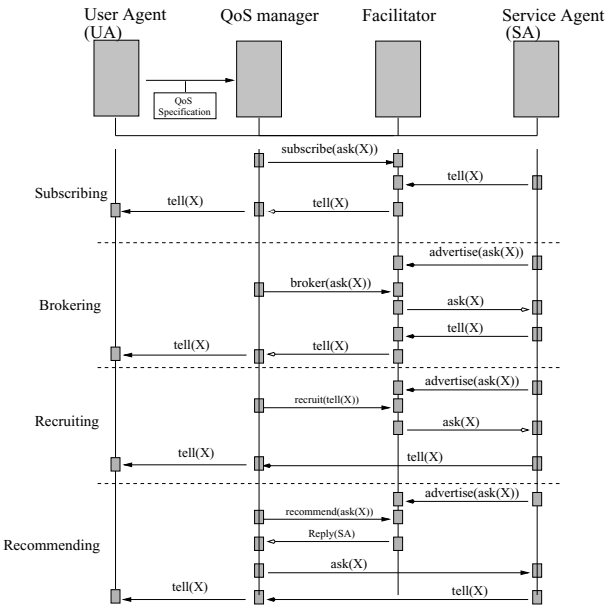


Fig. 2. Facilitator mediated QoS negotiation

illustrated in figure 2 uses the services of special agents (facilitators) that act as brokers between agents in need of some service and other agents that provide them. Mediation involves on one hand, needy agents subscribing to services and on the other hand facilitators brokering, recruiting, and recommending agents that registered their identities and capabilities.

### 3 Multi-agent System-Based QoS Negotiation

#### 3.1 The Problem

In standard QoS provision schemes for application running on small or local area networks, a QoS manager determines all configurations that can sustain an activity by:

- identifying necessary system components and building potential configurations,
- classifying these configurations, and
- selecting the most suitable configuration.

This approach assumes that the QoS manager has knowledge of potential service providers, system components and resources that exist in its environment and can communicate directly with them. As long as the number of entities involved in this service is small, this scheme is feasible and communication costs are acceptable. However, in a heterogeneous setting like the Internet with millions of computers, this approach shows two clear limitations:

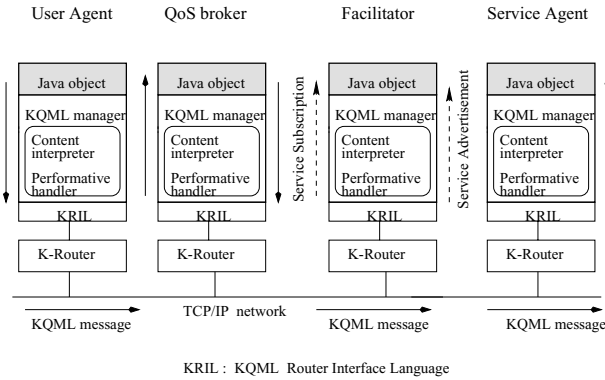
- *First:* During negotiation, the QoS manager alone must bear all the burden of identifying and selecting appropriate resources on a large scale networks like the Internet. This situation adds a substantial overload on the operation of the QoS manager. In addition, services and resources may not be guaranteed consistently.
- *Second:* When the number of entities involved in a direct communication with the QoS manager is modest, communication costs remain reasonable. However, in the Internet, these costs become prohibitive even with auxiliary local QoS managers.

To address these scalability and communication costs issues, we propose a framework for QoS negotiation illustrated in figure 3 where clients applications and service providers engage in a mediated communication. The mediators called *facilitators* and *QoS brokers* are supplied with information about identities and capabilities of service providers by the providers themselves. These entities are modeled as software agents with attributes, capabilities and mental attitudes as in AI. At the core of our framework lies the concept of multi-agent system composed of a user agent, a QoS manager agent, a facilitator agent, and service agents (network agents) communicating in KQML.

#### 3.2 Concepts and Framework Description

##### Concepts :

Prior to starting a service, a user specifies and supplies the QoS manager with a level of service expressed in QoS parameters. Then, the QoS manager must



**Fig. 3.** System architecture

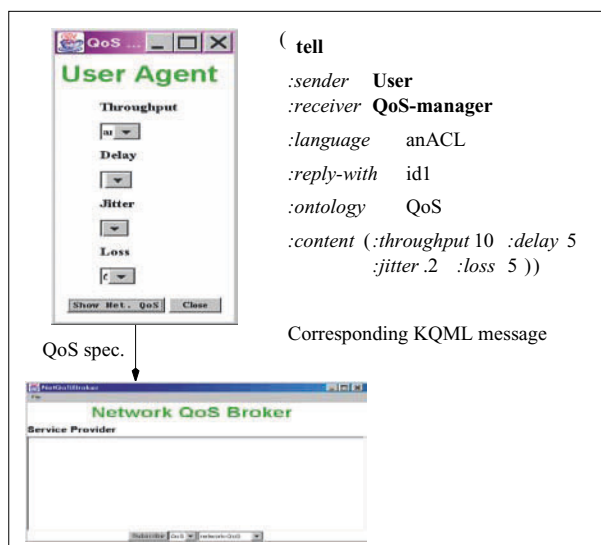
identify the set of components that can sustain this service. This process uses the following concepts:

- An ontology provides a vocabulary for representing and conveying knowledge about a topic (e.g. QoS) and a set of relationships that hold among the terms in that vocabulary. Our architecture uses four ontologies:
  - \* a *yellow page* ontology for service advertisement by service agents,
  - \* a *white page* ontology for finding the location of an agent given its name,
  - \* a *general QoS* ontology for the current domain knowledge,
  - \* and a *QoS broker* ontology for asking network options by the user and QoS broker.
- A *KQML manager* encompasses:
  - \* *Conversations* that group messages with a common thread identified by the “:reply-with and :in-reply-to” parameters;
  - \* *content interpreters* that handle incoming and related response messages according to the ACL, content language and ontology associated to these messages;
  - \* *performative handlers* that process a message performative in conjunction with its ACL, content language and ontology.

### QoS Negotiation Protocol :

In our framework, four types of agents communicate in KQML according to the following protocol:

- The user informs its agent via an interface of the required level of service.
- The user agent sends to the QoS manager agent a KQML message with required levels of service expressed in appropriate QoS parameters like in figure 4.
- The QoS manager needs to identify all components necessary to build a configuration that can sustain an activity. For this purpose, its agent sends a KQML message to the facilitator agent and can ask its cooperation in four different ways (subscription, brokering, recruiting and recommendation) in discovering all the



**Fig. 4.** User and QoS Broker interaction

appropriate resources. A structure of this KQML message and agent interaction is shown in figure 5.

- The facilitator agent acts as a resource broker that
  - \* recruits, recommends appropriate service agents (application, system, and network agents) to the QoS manager;
  - \* forwards the QoS manager messages (brokering and recruiting) to suitable service agents; and
  - \* informs (on subscription) or recommend to the QoS manager service agents that fulfill its requirements.
- All service agents (network agents) advertise their capabilities to the the facilitator agent upon registration. Upon request from QoS broker, the facilitator supplies the identities and locations of necessary network resources. At last, the user may view on an appropriate interface the available resources.

This QoS negotiation model for large scale Internet applications is applied in two ways: locally or remotely. When the required resources are available locally and registered at the local facilitator, negotiation is done at the current host as illustrated in figure 6.

On the other hand, when some resources are unavailable on site, the local facilitator reaches out to other facilitators at different locations as illustrated in figure 7. The local facilitator forwards requests (*broker-all*) to remote facilitators which in turn conduct a local inquiry. In fact, this approach to agents' interaction is already used in the field of *agent-based software engineering* where application programs are modeled as software agents and interaction is supported by an appropriate ACL. In this approach, agents are organized in a *federated system* with messages relayed by facilitators between hosts.

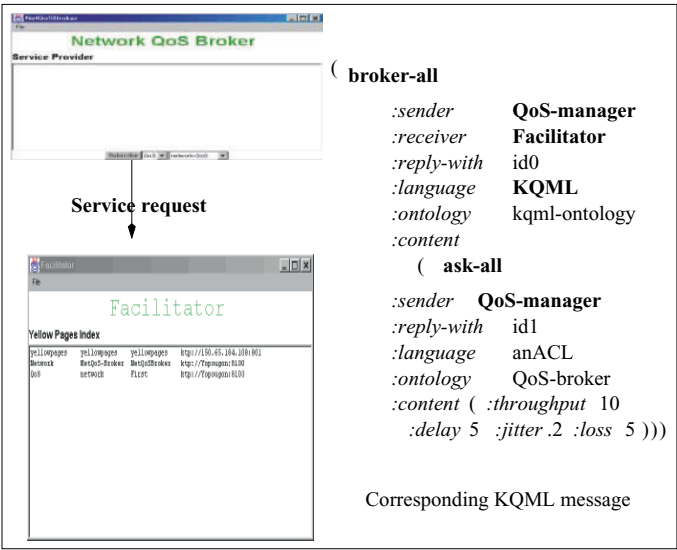


Fig. 5. QoS Broker and Facilitator interaction

3.3 Implementation

In experimenting with this model of resource discovery and QoS negotiation, we designed a prototype in the JAVA language to simulate QoS negotiation between several agents at the network level. That is to say, to illustrate our approach, a user agent and network agent communicate via a QoS broker and a facilitator in terms of network parameters only. First, local negotiation is considered, then

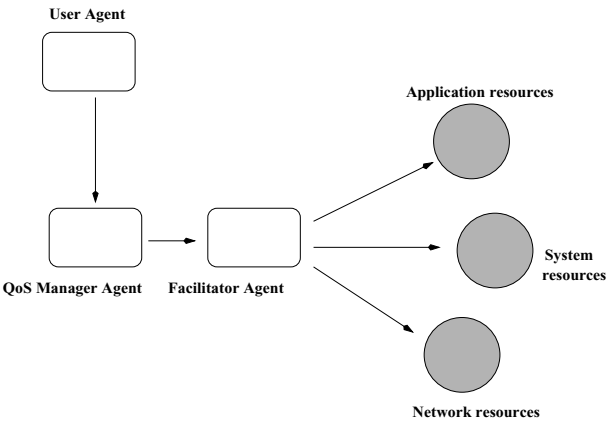
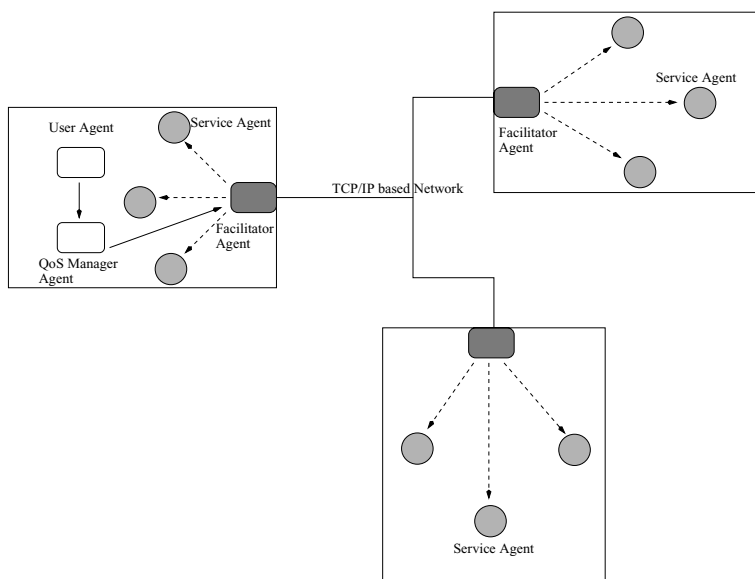


Fig. 6. Local QoS negotiation with a single facilitator dealing with resources inside a given host.



**Fig. 7.** Large scale QoS negotiation with several facilitators involved in the negotiation process across the Internet.

it is extended to remote locations across the Internet.

The implementation of our prototype includes the following tools:

- The Java-based KQML API called *JKQML* in [9]. The JKQML API with its structure in figure 8 adapted from [9] provides a platform for designing KQML-enabled agents. JKQML is based on the JAVA language and provides interoperability to software that needs to exchange information and services.

Handling KQML messages involves the following steps:

1. Instantiating a KQML manager with the method:  

```
public KQMLManager(String agentName, String protocol, int port);
```
2. Managing protocol handlers with the method:  

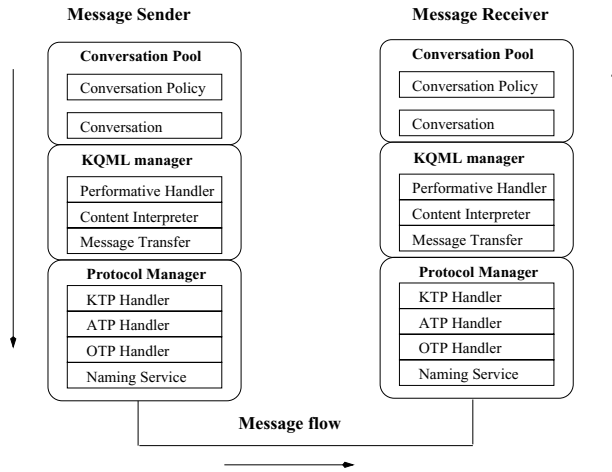
```
public void addProtocol(String protocol, int port);
```
3. Managing content interpreters with the method:  

```
public void addContentInterpreter(String acl, String language, String ontology);
```
4. Managing performative handlers with the method:  

```
public void addPerformativeHandler(String acl, String language, String ontology, String performative, PerformativeHandler ph);
```
5. Managing conversation termination with the method:  

```
public void setConvCleanupHandler(ConvCleanupHandler c).
```

- We used the Stanford KSL *Ontolingua* ontology editor at [8] to design both the general QoS ontology and the QoS-broker ontology used by the language interpreter. Then, we extended our simulation program to a larger TCP/IP network with facilitators at different locations communicating in KQML. A couple of



**Fig. 8.** Structure of JKQML

networks with different characteristics (throughput, delay, jitter, and loss) were discovered successfully and displayed on a local user interface. Figure 5 and figure 4 illustrate some transactions between the participating agents.

## 4 Issues and Perspectives

In an open and heterogeneous environment like the Internet, agents that interact and coordinate their activities face some major challenges:

- How can they find one another and specially, locate the facilitators? As the number of facilitators grows, finding their location becomes a real concern. The idea of introducing a *facilitator directory* that forwards external inquiry to all facilitators across the Internet could address this problem.
- Although many ACLs exist today, the communication language chosen should express concisely the message content of an agent. That is to say, the message semantics of an ACL must be consistent across platforms.
- With any kind of message transport protocol (KQML transport protocol (*ktp*) or agent transport protocol (*atp*)), the issue of fault tolerance due to network failure remains. Multi-agent systems must rely on a robust and reliable environment. However, the heterogeneous nature of the Internet offers no guaranty.

In addition to negotiation on the network layer, we are looking forward to extending our model to the application and system layers as well. This way, with a suitable QoS translation scheme between these layers, it is possible to cover a complete end-to-end QoS negotiation.

We intend to investigate the alternative of mobile agents as a message transport protocol. Enabling facilitators to move around the network, deliver information and collect advertisements like mobile agents is an option we are interested in. These *mobile facilitators* can interact on site with local QoS brokers and ser-

vice agents. In addition, as the new Foundation for Intelligent Physical Agents (*FIPA*) ACL standard is emerging, we are looking forward to implement our model in this language.

## 5 Related Work

A number of service discovery protocols have been implemented for different platforms. Some examples are the Service Location Protocol (SLP) designed by the Internet Engineering Task Force, the Dynamic Host Configuration Protocol (DHCP), the CORBA architecture [7] with its *Trader* and *Naming Services*, and recently Sun Microsystems's Jini.

### 5.1 The Service Location Protocol (SLP)

The idea of using multiple agents for the discovery of services across a local area network has already been used by the SLP. In this model, a user agent (UA) acts on behalf of a user or client in need of a service while a service agent (SA) declares its services to a directory agent previously discovered. In addition, a directory agent (DA) accepts requests and registrations from a UA or a SA.

There are two fundamental differences between the SLP scheme and our approach: SLP uses multicast and DHCP protocols to initialize its scalable service discovery framework. However, as DHCP cannot extend to the entire Internet, SLP is unable to scale to the entire Internet. A user agent itself must send its queries to a remote DA when a service is not available locally. In contrast, our approach considers a federation of services as illustrated in figure 7 with several facilitators. Only facilitators may forward requests from one region to another. In addition, we use KQML messages to convey these requests across the Internet.

### 5.2 The CORBA Trader and Naming Services

CORBA is a middleware that enables a *client application* to request information from an *object implementation* at the server side. In addition, CORBA can advertise available objects and services on behalf of object implementations via a *Common Object Services Specifications* (COSS) service called the *Trader Service*. Services are registered with the *Naming Service* by specifying its name and object reference. A client who wishes to access the service specifies the name of the service, which the Naming Service uses to retrieve the corresponding object reference. Whereas services are registered with the Trader Service by specifying its service type, properties and object reference. A client who wishes to access the service, specifies the type of the service and constraints. Therefore, the Trader Service can be viewed as a yellow pages phone book.

In spite of the similarities in both approaches, it is important to note that the main difference between our system and CORBA services is that we are dealing with messages which bear meaning and are organized in conversations. The players in our system are agents that are engaged in structured conversations



while CORBA enables applications to exchange only objects, data structures, and propositions.

### 5.3 Jini

Jini is a network operating system by Sun Microsystems aimed at a broad range of electronic devices and software services assembled in a single distributed computing space. Although the components work together to serve a common goal, they're still identified as separate components on a network. The Jini discovery architecture is similar to that of SLP. Jini agents discover the existence of a Jini Look Up Server, which collects service advertisements like the facilitators in our system. Jini agents then request services on behalf of client softwares by contacting the Look Up Server.

## 6 Conclusion

In this paper, we have presented a framework for resource discovery and quality of service negotiation over the Internet. The main point is that our framework relies on the concept of multi-agent systems and agent communication language. In contrast to automatic resource discovery protocols like the SLP, our scheme scales to the entire Internet. To illustrate its effectiveness, we designed a prototype based on the IBM Java KQML API with several agents: user agent, QoS broker agent, facilitator agent, and network agents that interact in the KQML agent communication language. Although this approach may look attractive, its main drawback lies in the important number of facilitator agents that the system must deal with. In the future, we intend to let these facilitators move from host to host with information just like mobile agents.

## References

1. Droms R.: rfc1541. Technical report, IETF, Network Working Group, <http://www.cis.ohio-state.edu/htbin/rfc/rfc1541/.html>, October 1993.
2. Genesereth R. Michael, and Ketchpel P. Steven: Software agents. *Communications of the ACM*, 37:48, July 1994.
3. Patil, Ramesh S. , Fikes, Richard E.: The DARPA knowledge sharing Effort: Progress Report, In: Michael Huhns and Munindar P. Singh (Ed.), *Readings in Agents*, Morgan Kaufmann, 1998, pp 243-254.
4. Perkins C.: SLP white paper, Technical report, Sun Microsystems, <http://playground.sun.com/srvloc>, 1998.
5. Keith W. Edwards.: *Core Jini*, Sun Microsystem Press, June 1999.
6. Kone Tadiou Mamadou, Akira Shimazu, and Tatsuo Nakajima : The State of the Art in Agent Communication Languages. (submitted) to *Knowledge and Information Systems*, 1999.
7. Randy Otte, Paul Patrick, Mark Roy : *Understanding CORBA, The Common Object Request Broker Architecture*, Prentice Hall, 1996.
8. Stanford KSL Network Services : *Ontolingua*, ontologies editor. <http://www-ksl-svc.stanford.edu:5915/>.
9. Tsuchitani Hajime and Furusawa Osamu : JKQML. *AlphaWorks*, IBM, 1998.

# Benchmarking of Signaling Based Resource Reservation in the Internet

István Cselényi<sup>1</sup>, Gábor Fehér<sup>2</sup>, and Krisztián Németh<sup>2</sup>,

<sup>1</sup>Department of Network Services, Telia Research AB  
*Vitsandsgatan 9, S-12386 Farsta, Sweden, Phone: +46 8 713 8173*  
istvan.i.cselenyi@telia.se

<sup>2</sup>High Speed Networks Laboratory,  
Department of Telecommunications and Telematics, Technical University of Budapest  
*Pázmány Péter sétány 1/D, H-1117 Budapest, Hungary, Phone: +36 1 463 3119*  
{feher, nemeth\_k}@ttt-atm.ttt.bme.hu

**Abstract.** This paper investigates the scalability limitations of IP resource reservation protocols using RSVP and Boomerang as examples. The memory and processing time consumption of signaling message primitives were measured as a function of the total number of concurrent reservation sessions on PC-based routers running Linux and on a commercial router. The signaling handling algorithm of the implementations were analyzed as well and critical operations were identified. Our results show that CPU time is a more significant scalability concern than the router memory and also that the former is very dependent on the implementation and complexity of the signaling algorithm. Thus the same Linux PC can handle Boomerang reservation requests several hundred times faster than RSVP requests.

## 1 Introduction

It is not so unpopular any more to mention signaling in the context of Differentiated Services [1] as it was some years ago [2]. Aggregation of reservation sessions, flow identifiers and signaling messages give a realistic hope in demolishing scalability limitations. However, there remain still some points in the network where micro-flows shall be differentiated from each other (e.g. border nodes) and therefore it is important to scrutinize the scalability limitations of per flow resource reservations in such network nodes.

There are several factors, which influence scalability. We investigated the resource demand in signaling-aware routers in terms of memory and processing power for the RSVP [3] and Boomerang [4] resource reservation protocols, which have working implementation for measurements and public source code. However, the same benchmarking framework could be applied to other resource reservation protocols, such as ST-II [5], Yessir [6], Ticket [7] or DRP [8]. A summary of resource reservation protocols and comparative evaluation can be found in [8].

## 1.1 The RSVP Protocol

The Resource reSerVation Protocol (RSVP) is a receiver oriented signaling protocol, which initiates soft reservation states in network nodes, which are on the path of the (possible multicast) datagram transfer. There are two main signaling messages defined in RSVP, both are sent end-to-end between the sender and the receiver host(s). The *Path* message is issued by the sender host and forwarded hop-by-hop towards the receiver(s). *Path* registers the address of previous hop at each node, conveys the sender's traffic specification and optionally network specific information from involved nodes. By sending a *Resv* message as a reply to the received *Path*, the receiver initiates a reservation setup. Since the previous hops are captured in each hop thanks to *Path*, the *Resv* message travels on the same route as the *Path* even for asymmetrical routes. *Resv* specifies the amount of resources to reserve in each router en route. *Path* & *Resv* messages are also used for refreshing the soft reservation states in the routers, which are otherwise removed after a certain time.

There are three more basic RSVP messages. The *Path Tear* and *Resv Tear* messages are used to tear down a certain reservation state, while the *Resv Conf* message is used for acknowledging the receiver that the reservation was successful.

## 1.2 The Boomerang Protocol

Boomerang is a lightweight, sender oriented IP resource reservation protocol. Since it is described in details in [4][9][10] just a short overview is given here.

The Boomerang protocol can be used for specifying and reserving network resources for uni- or bi-directional traffic streams between two IP nodes. Traffic stream can mean a single data flow or a flow aggregate (e.g. DS behavior aggregate or flows between subnets), where up- and downstream routes can differ. Reservation setup is made by a single message loop (currently the ICMP Ping message) and kept alive by periodically repeated Boomerang messages, which establish soft reservation-states (see Fig. 1).

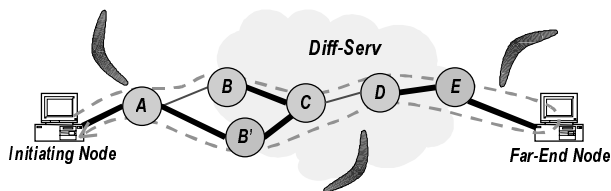


Fig. 1. Reservation Setup with the Boomerang Protocol

Unlike in RSVP, the same protocol message is used for refreshing or triggering changes in a reservation state or tearing it down. The Initiating Node that is responsible for keeping track of the reservation session periodically generates signaling messages. Boomerang messages are forwarded hop-by-hop in the network

up to the Far-End Node, from where they are returned back to the Initiating Node. The required service quality can be given in various ways; by specifying a DSCP [12] and peak information rates for the forward and backward directions or by providing an IntServ-like object [13]. In our prototype implementation [4] Boomerang protocol messages are wrapped into *ICMP Echo / Echo Reply*-s which are supported in the majority of network nodes and hosts, giving good chance for an easy penetration of the protocol in current Internet.

This paper is organized as follows: Section 2 describes our benchmarking framework by defining the investigated factors and the measurement scenario. Section 3 provides the measurement results and their analysis. Finally, conclusions are drawn in Section 4.

## 2 Benchmarking Framework

### 2.1 Router Resources

The main scalability concern opposed to signaling-based resource reservation at microflow scale is that this paradigm places a load on signaling-aware routers, which is proportional to the number of reserved flows. This load can be quantified by measuring the *processing time* and *memory requirement* per signaling message types as a function of concurrent reservation sessions. The following equations can be derived from simple heuristics:

$$t = \tau + \alpha n, \quad n < N_t \quad (1)$$

where  $t$  is the message processing time;  $n$  denotes the number of reserved flows in the router, while  $\tau$  and  $\alpha$  are message specific constants and  $N_t$  represents the scalability limit above which the linear approximation is not valid. With similar notions, the memory demand can be modeled with the following equation:

$$m = \mu + \beta n, \quad n < N_m \quad (2)$$

The constants  $\tau$  and  $\alpha$  are larger in case of a more complex signaling handling algorithm, thus it is worth to analyze the related source code and perform the tests for each signaling primitive separately. On the contrary, constants  $\mu$  and  $\beta$  are independent from the atomic signaling primitives and they are determined solely by the memory allocation scheme of the implementation and size of reservation states.

Other performance metrics of the reservation protocol, such as *reservation setup time*, *signaling overhead* on links and *number of signaling messages per reservation* can be retrieved from simple calculations by using the results of these measurements and the protocol specifications as input [9].

2.2 Critical Handler Operations

We identified three critical operations in the signaling handling software (referred to as handler), which is running in signaling-aware routers.

Every time, when a signaling-aware router receives a signaling message, it should check in the record of existing reservation sessions whether the message refers to a new flow, triggers some change in an existing reservation or just refreshes it. The performance of this *reservation session lookup* operation depends on the data structure used for storing the reservation states and the algorithm that looks up the stored entry.

If the signaling message refers to a previously unknown reservation, the handler must *create a new reservation state* (i.e. new entry). The latency of this operation depends on the size of the entries that the handler has to fill out and on some additional procedures, like initializing refresh timers.

The signaling handler has to *set up the internal traffic control* mechanisms in the router in order to maintain the QoS of each individual reservation. This operation is not performed for refresh messages. Although the initiation of traffic conditioners includes setting of the shaper, meter or dropper, we concentrated only on setup and parameterization of the queues in the routers.

2.3 Measurement Scenario

Measurements were carried out to obtain the memory and processing time consumption of the different protocol primitives. Resource reservation was performed in the Router, which connected Host 1 and Host 2 (Fig. 2). Signaling messages on ingress and egress ports of the router were intercepted and logged together with time stamps by Sniffer running *tcpdump*, which was connected to both sides of the router via hubs. Pentium II PCs were used with 300 MHz CPU and 64 MB RAM, running Linux as operating system with a kernel version that supports QoS [14].

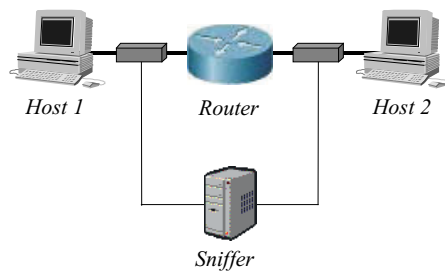


Fig. 2. Measurement Scenario

Since the performance of signaling handling very strongly depends on the actual implementation, we have examined three different QoS routers running two kinds of reservation protocols. The three router configurations are shown in Table 1.

**Table 1.** Investigated Router Configurations

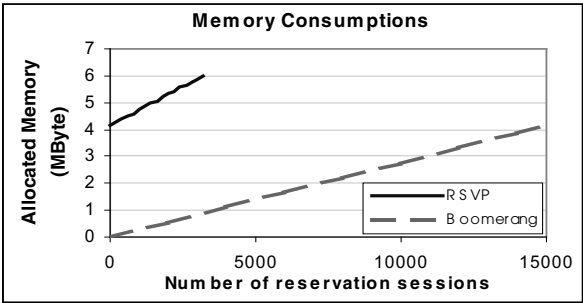
Configuration	Hardware	Software	Protocol
SW-RSVP	PC	Linux, user space	RSVP
SW-Boomerang	PC	Linux, kernel space	Boomerang
HW-RSVP	Dedicated	No data available	RSVP

We used Telia’s prototype implementation and source code for SW-Boomerang and the source code of the public ISI implementation for SW-RSVP [15] measurements. The dedicated hardware was a 3Com CoreBuilder 3500 router, which supports RSVP. We repeated each measurement 30 times. The goal was to benchmark the load caused by the signaling, so there was no data traffic utilizing the reserved resources: only setup, tear down and the periodically sent refresh messages loaded the router.

**3 Results**

**3.1 Measurement of Memory Consumption**

In order to determine the memory usage versus the number of concurrent reservations, we set up an increasing number of reserved flows and measured the amount of memory that is allocated in case of SW-RSVP and SW-Boomerang. We could not measure the allocated memory in case of HW-RSVP, having no access to the internal parts of the router. The measurement results are shown in Fig. 3.



**Fig. 3.** Memory Consumption

The memory allocation measurements revealed that the memory consumption is linear in case of both protocols. However, while SW-Boomerang takes 288 bytes, SW-RSVP needs 614 bytes to record the details of one reserved flow, partially because

they use different traffic descriptors. Notice that SW-RSVP's initial memory consumption ( $\mu$ ) is higher than SW-Boomerang's, because latter is a kernel module and takes its initial memory allocation from the kernel space, while SW-RSVP is a user mode process with large arrays and a built-in application programming interface.

It is also visible in the figure that the investigated implementation of RSVP does not scale above to 3200 reservation sessions. This limitation was due to our receiver endpoint, which could not handle more. The figure shows the result of the first 15,000 reservation sessions for SW-Boomerang, however we measured up to 60,000 flows where the curve still had the same linear behavior.

### 3.2 Measurement of Message Processing Time

The time interval elapsed between the arrival and departure of a signaling message in the router was measured by capturing the signaling traffic before and after the router, and logging the absolute time. Apart from the time of actual message processing, this interval includes the time, which the packet spends in the forwarding plane, e.g. the time of classification and routing. These additional factors are expressed in the  $\tau$  constant (see equation 1).

For measuring the processing time as a function of established reservations, we set up the desired number of reservations and then launched a test program, which set up and shortly afterwards tore down one test reservation, repeating this 30 times with a longer pause among the setup and tear down pairs. RSVP refresh messages were not measured, because they are generated by neighboring RSVP routers, so an incoming refresh message does not immediately leave on the outgoing port, instead it is sent out when the router decides that the connection requires it.

Table 2, Table 3 and Table 4 show our main results, where we characterized the measured processing time values with their median value  $M$  and with the scaled coefficient of variation ( $SCV$ ).

We can see that the processing time and the SCV always increase proportionally to the number of maintained reservations in the router, but different protocols and implementations yield different slopes. The most interesting result is that the SW-Boomerang prototype can process one Boomerang reservation message within 53-59 microseconds, while the SW-RSVP implementation needs more than 15 ms, 8 ms for handling an RSVP Path, RSVP Resv message, respectively. Moreover, in case of SW-Boomerang the scale of reservation sessions is larger by two orders of magnitude, than in case of the two RSVP implementations (i.e. 60000 and 600, respectively).

### 3.3 Analysis of Algorithmic Complexity

We have analyzed the available source code in order to estimate the time the router spends on critical operations

**Table 2.** Measured Processing Time in case of SW-Boomerang

Message Type	Number of Sessions / Median and SCV							
	0		20 000		40 000		60 000	
	M (ms)	SCV	M (ms)	SCV	M (ms)	SCV	M (ms)	SCV
Boomerang (Reservation)	0.053	8.963 E-03	0.056	7.801 E-03	0.057	6.879 E-02	0.059	1.501 E-01
Boomerang (Reservation Tear)	0.054	2.248 E-02	0.057	6.178 E-03	0.057	8.616 E-02	0.058	1.647 E-01

**Table 3.** Measured Processing Time in case of SW-RSVP

Message Type	Number of Sessions / Median and SCV							
	0		200		400		600	
	M (ms)	SCV	M (ms)	SCV	M (ms)	SCV	M (ms)	SCV
RSVP Path	15.44	9.90 E-07	15.57	1.601 E-05	15.72	3.102 E-05	15.89	3.458 E-05
RSVP Path Tear	0.150	2.22 E-03	0.151	1.052 E-01	0.151	1.650 E-01	0.156	4.095 E-01
RSVP Reservation	8.607	2.59 E-06	9.081	1.939 E-05	9.583	1.996 E-05	10.06	3.938 E-05
RSVP Reservation Tear	8.078	4.58 E-06	8.553	4.124 E-05	9.025	3.399 E-05	9.521	1.196 E-04
RSVP Confirmation	0.243	9.67 E-04	0.257	1.450 E-02	0.241	4.276 E-02	0.240	1.388 E-01

**Table 4.** Measured Processing Time in case of HW-RSVP

Message Type	Number of Sessions / Median and SCV							
	0		200		400		600	
	M (ms)	SCV	M (ms)	SCV	M (ms)	SCV	M (ms)	SCV
RSVP Path	7.402	5.389 E-01	12.26	6.690 E+00	20.46	2.544 E+00	577.3	8.814 E-01
RSVP Path Tear	5.554	9.932 E-01	5.455	9.800 E+00	6.682	3.698 E+00	216.1	1.554 E+00
RSVP Reservation	11.15	5.359 E-01	13.41	1.508 E+00	16.83	3.440 E+00	262.8	1.259 E+00
RSVP Reservation Tear	8.342	1.781 E-01	18.07	6.275 E+00	35.61	2.975 E+00	70.49	4.778 E+00
RSVP Confirmation	5.385	1.170 E+00	5.366	6.903 E-02	5.448	1.644 E+01	43.30	2.008 E+00

### Reservation Session Lookup

According to the source code the *bucket hash* algorithm is used in ISI's RSVP implementation, while *modified binomial tree* is implemented in Boomerang prototype. We characterized the efficiency of these lookup algorithms by comparing processing latency of different message primitives. Investigating the source code of the ISI RSVP implementation, we found that the *RSVP Path Tear* message processing should be the fastest, because it contains solely a session entry lookup and delete. The



processing time of *Path Tear* message begins around  $150\mu\text{s}$  and slightly raises with the number of reservations, while the variance of the measured values increases steeper as it can be seen on Fig 4. This phenomenon could be explained by the increasing size of the hash table.

In case of SW-Boomerang, we investigated the reservation tear message, which is functionally the same as the corresponding RSVP message, although it cleans the associated queue as well. The result indicates that processing time of a Boomerang tear message starts at  $53\mu\text{s}$  and increases very slowly. In case of 60 000 reserved sessions, the reservation tear takes just  $58\mu\text{s}$ . The variance of processing time did not show a noticeable change while we increased the number of sessions.

It is worth to mention that the measured processing time is affected by other factors as well, for instance the kernel performs numerous operations on the received packet while it gives to the protocol handler routines. Thus we measured the simple forwarding time too, where there was no QoS protocol running on the router. We found that forwarding a single 100 bytes-long data packet takes about  $40\mu\text{s}$ .

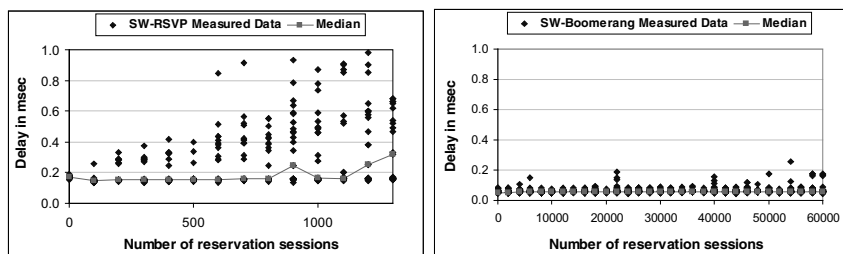


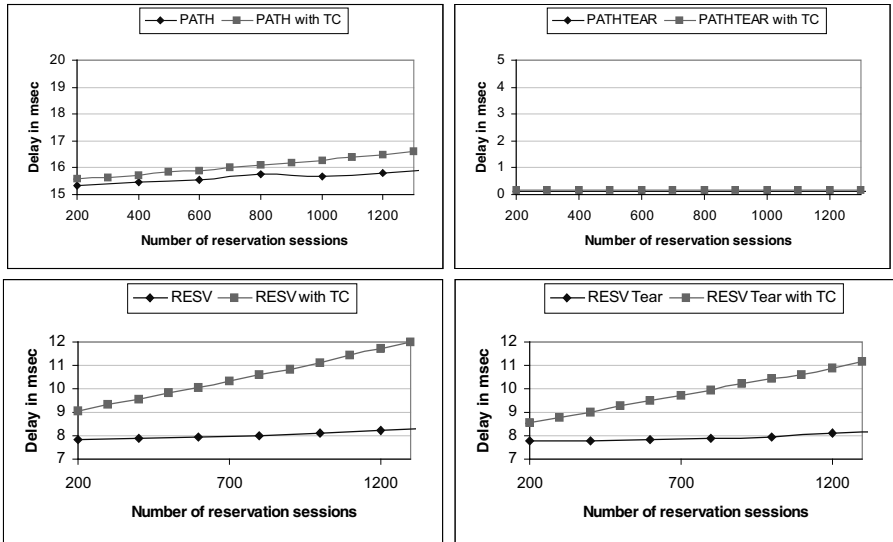
Fig. 4. Tear-down Messages for SW-RSVP (Path Tear) and SW-Boomerang

Apart from the faster execution of kernel space programs (SW-Boomerang), the reason of unequal lookup performance is the difference in data structure and lookup algorithm. SW-Boomerang takes smaller structures and uses a binomial tree lookup algorithm, which costs about  $13\mu\text{s}$  per flow (since packet forwarding time is  $40\mu\text{s}$ ), and increases very slowly. The bucket hashing algorithm and large data structures used by RSVP results in larger processing time, around  $110\mu\text{s}$  without packet forwarding, and it increases faster than for Boomerang.

### Internal QoS Setup

The protocols differ in the implementation of the traffic control as well. While SW-RSVP uses the operating system's built in traffic control routines, the SW-Boomerang uses own traffic control implementation that is bounded to the protocol.

We repeated our SW-RSVP measurements with disabled traffic control operation and estimated the time spent for internal QoS setup. The difference in the message processing times expresses the time consumed by maintaining traffic queues. Fig. 5 shows the median values for reservation and path related RSVP messages with and without traffic control.



**Fig. 5.** RSVP Message Processing Time with and without Traffic Control

We can observe that in case of *Path* and *Path Tear* messages, there is no significant difference between the two measurement series. *Path* message is affected a little bit more than *Path Tear*, because RSVP fills out an advertisement message block that reflects the router's load calculated from traffic parameters. However in case of reservation messages, a strong difference can be revealed. The processing time was increased with 1ms (i.e. almost 15 %) and the slope of the line is larger than without traffic control. We can confirm it looking up the source code, as both types of reservation message calls the traffic control interface and the raising slope can be explained with the increasing number of queues that must be maintained.

In case of SW-Boomerang, the handler uses its own traffic control routines what simplifies the internal QoS setup thus the related CPU time is marginal. On the other hand, it should be noted that the current prototype implementation of Boomerang deals with much simpler traffic descriptor (peak rate only) than RSVP.

### Creation of New Reservation State

According to the source codes, new reservation states are created in case of RSVP *Path* and *Resv* messages. Unlike this, the Boomerang handling prototype stores reservation states in lookup entries and no state information is required for handling a state machine of signaling.

We tried to make deductions to this fact from the measured processing time values. RSVP *Path* and *Path Tear* messages show the biggest difference, which is around 15ms (see Fig. 6). When we inspected the source code, it clearly showed that the difference arises directly from the creation of a new reservation state. The rest of the functionality, like reservation session lookup, can be found in both message handling routines.

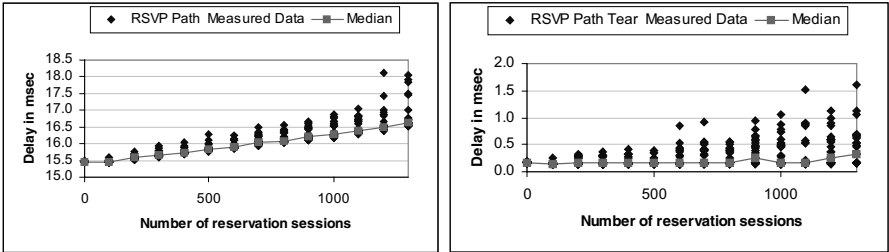


Fig. 6. RSVP Path and Path Tear Message Processing Time

3.4 Hardware RSVP Measurements

In case of HW-RSVP, we were restricted to processing time measurements, as we had no access to the internal parts of the router. The processing time measurements are presented in Table 4 and surprisingly show that HW-RSVP performs worse than SW-RSVP (see in Table 3).

In our experiments the device was not capable to scale above 600 reservation sessions. This was not a built-in limit, but above that the router’s RSVP handler was not able to refresh the flows, meaning that it was not able to generate refresh messages.

If we assume that the *RSVP Path Tear* message, displayed in Table 4, mainly consist of the session lookup procedure, as we have shown it in case of SW-RVSP, then we can claim that the router device had an inefficient session lookup routine. It is thought that the device was not designed for more than few hundred sessions. We can confirm that the most time consuming part is the session lookup, during processing the *Path Tear* message, if we compare results of *Confirmation* message and assume that there is no session lookup for *Confirmation* messages in this implementation. We could not switch off the traffic control routines in the 3COM router thus could not retrieve the time factor related to queue handling. As the measured *Path* message processing time shows, it begins from a lower value than SW-RSVP, but it increases much faster.

4 Conclusions

This paper investigated different factors, which contribute to the scalability limitations of signaling based resource reservations in Internet.

We have shown that the reservation lookup scheme is an important factor in scalability. Lookup of reservation session is three times faster for SW-Boomerang than for SW-RSVP and this factor is even worse for HW-RSVP. Another result of the measurements is that the reservation state creation is the most time consuming event in case of SW-RSVP. It takes about 15ms, while in case of SW-Boomerang the cost of

reservation state creation is just 53  $\mu$ s. We measured the time factor of traffic control in case of SW-RSVP and SW-Boomerang. The results indicate that internal implementation of traffic queues can speed up the processing of related messages, moreover it can decrease the impact of established reservation session on processing time. We observed that the HW-RSVP implementation has worse performance characteristics than SW-RSVP. The measured processing time of protocol messages goes exponential with the number of reservation sessions; meanwhile the scaled coefficient of variation is larger by four orders of magnitude than in case of SW-RSVP.

Additional measurements show that signaling intensity – the number of atomic signaling messages received by the router per second – is also an important scalability factor. In case of SW-Boomerang we could scale up to 3600 messages per second without having lost signaling messages. On the other hand SW-RSVP has been saturated already by 30 messages per second.

It should, however, be noted that apples cannot be fairly compared to pears; therefore the difference of configuration should not be neglected while looking at our results. The SW-RSVP uses complex traffic descriptors and the handler run as a Linux user-space routine. On the other hand, the SW-Boomerang prototype handles only peak rate reservation running as a Linux kernel module.

## Acknowledgements

We would like to thank the excellent design and implementation work of Joakim Bergkvist and the help in the measurements to Tomas Engborg, David Ahlhard and Norbert Végh, all from Telia Research Sweden. We are also grateful for the help of members of the RSVP mailing list, who provided us with useful information about the implementation of RSVP protocol.

## References

1. S. Blake, D. Black, M. Carlson, E. Davies, Z. Wang, and W. Weiss, "An Architecture for Differentiated Services", RFC 2475, December 1998.
2. Y. Bernet, R. Yavatkar, P. Ford, F. Baker, L. Zhang, M. Speer, R. Braden, B. Davie, J. Wroclawski, E. Felstaine, "A Framework for Integrated Services Operation Over Diffserv Networks", Internet Draft, September 1999, <draft-ietf-issll-diffserv-rsvp-03.txt>
3. R. Braden, L. Zhang, S. Berson, S. Herzog, and S. Jamin, "Resource Reservation Protocol (RSVP) Version 1 Functional Specification", IETF RFC 2205, Proposed Standard, September 1997.
4. D. Ahlhard, J. Bergkvist, I. Cselényi, "Boomerang Protocol Specification", Internet Draft, June 1999; <http://www.ietf.org/internet-drafts/draft-bergkvist-boomerang-spec-00.txt>
5. C. Topolcic, "Experimental Internet stream protocol, version 2 (ST-II)", Request for Comments (Experimental) RFC 1190, Internet Engineering Task Force, October 1990.
6. P. Pan, H. Schulzrinne, "YESSIR: A Simple Reservation Mechanism for the Internet", <http://www.ctr.columbia.edu/~pan/work/yessir/yessir.ps>

7. A. Eriksson, C. Gehrmann, "Robust and Secure Light-weight Resource Reservation for Unicast IP Traffic", International WS on QoS'98, IWQoS'98, May 18-20, 1998
8. Paul White and Jon Crowcroft, "A Case for Dynamic Sender-Initiated Reservations in the Internet", Journal of High Speed Networks, Special issue on QoS Routing and Signaling, Vol 7 No 2, 1998.
9. G. Fehér, K. Németh, M. Maliosz, I. Cselényi, J. Bergkvist, D. Ahlhard, T. Engborg, "Boomerang – A Simple Protocol for Resource Reservation in IP Networks", IEEE WS on QoS Support for Real-Time Internet Applications, Vancouver, Canada, June 1999
10. I. Cselényi, G. Fehér, K. Németh, "On the Scalability of Scalability of Signaling-Based Resource Reservation in Internet", submitted to the IEEE Journal on Selected Areas in Communications
11. I. Cselényi, R. Szabó, "Service Specific Information Based Resource Allocation for Multimedia Applications", accepted for publication in journal of Informatica
12. K. Nichols, S. Blake, F. Baker and D. Black, "Definition of the Differentiated Services Field (DS Field) in the IPv4 and IPv6 Headers", RFC 2474, December 1998.
13. R. Braden, D. Clark, and S. Shenker, "Integrated Services in the Internet Architecture: An Overview", Internet RFC 1633, July 1994.
14. "Differentiated Services on Linux", Internet Draft, June 1999 <draft-almesberger-wajhak-diffserv-linux-01.txt>
15. University of Southern California Information Sciences Institute (USC ISI) implementation of RSVP: <http://www.isi.edu/div7/rsvp/>

# Kalman and Neural Network Approaches for the Control of a VP bandwidth in an ATM Network

Raphaël Feraud \*, Fabrice Clérot \*\*, Jean-Louis Simon, Daniel Pallou, Cyril Labbé, Serge Martin

France Télécom C.N.E.T., Technopole Anticipa, 2 av. Pierre Marzin, BP 40  
22307 Lannion Cedex France

**Abstract.** Allocating resources to data traffic in telecommunication networks is a difficult problem because of the complex dynamics exhibited by this kind of traffic and because of the difficult trade-off between the delivered quality of service and the wasted bandwidth.

We describe and compare the performances of two controllers of different designs (a Kalman filter and a neural network) to adaptatively control the bandwidth of a VP connecting two network nodes so as to keep the quality of service close to a given target.

Simulations are carried out on a hardware emulator which allows a fast and faithful simulation of the switch functionalities.

The Kalman filter obtains good performances and the original (and specific to this problem and its implementation constraints) neural network training strategy allows to obtain a faster and more accurate control.

## 1 Introduction

Managing data traffic in multiservice networks is a significant challenge for network operators since the data traffic volume is already equivalent to the classical telephony volume and is increasing at a much faster rate and it is known that data traffic is not well represented by classical traffic models, so that standard dimensioning techniques fail [4].

More precisely, the study of real data traffic traces [9], [5], [10] has shown that such traffic either is stationnary and should be modelled by processes with long range dependence and complex short time-scale dynamics [5] [10], or is not stationary and should be locally modeled by short range processes whose parameters have to be tracked on-line [9].

Dimensioning the network resources is a significant challenge in both cases: if the traffic is non-stationary, on-line tracking and prediction is necessary so as to allocate the resources; in the other case, static dimensioning from long range dependent models often leads to the allocation of very large resources protecting the network from very large deviations; such large deviations are

---

\* Email: raphael.feraud@cnet.francetelecom.fr

\*\* Email: fabrice.clerot@cnet.francetelecom.fr

however very rare and such static dimensioning wastes a lot of resources [3]. This is why, even assuming that traffic is stationnary and long-range dependent, on-line estimation of the parameters and prediction of the resources needed, that is adaptive dimensioning, seems a more promising approach.

In this work, we shall investigate two approaches for controller design; in the first approach, our a priori knowledge of the system is used to *select state variables* which sum up the state of the system as it may be observed (measured) and the controller design problem reduces to learn the relationship (“mapping”) between the state variables and the behaviour of the system. This approach is implemented by a neural network. In the second approach, our a priori knowledge of the system is used to *build a model* of this system depending on a few parameters and the controller design problem reduces to fit the model to the behaviour of the system. This approach is implemented by a Kalman filter.

Since this work is strongly constrained by implementation considerations, we shall first present in some details the mechanisms used to guarantee a minimum QoS and fairness in the network nodes. The design of the controllers is then described and the experimental environment and results are discussed.

## 2 The CMS switch

The CMS switch is an ATM switch developed at CNET with the aim of optimizing transfer of high speed data in an ATM network. It takes into account the specificity of data service: large semantic constraints and weak temporal constraints. A single connection-oriented data service is implemented in the backbone. In each switch, connections are isolated by the use of a per connection FIFO, and active connections share dynamically the bandwidth. Each connection is identified by its Virtual Connection (VC), its Minimum Guaranteed Rate (MCR), its Peak Cell Rate (PCR). The connections are multiplexed in a Virtual Path (VP) between two CMS nodes.

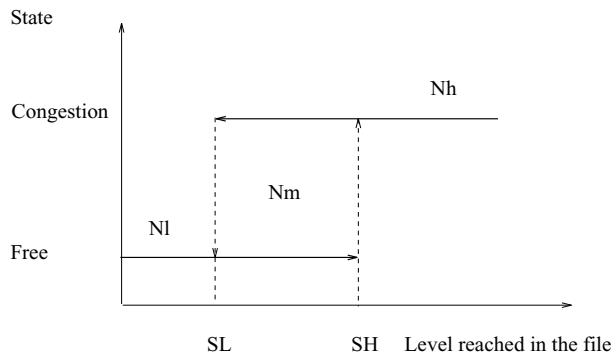


Fig. 1. The free/congested cycle. See text for explanations.

A hop by hop flow control is implemented between the transit nodes, ensuring the absence of packet losses in transit. The implementation of this hop by hop flow control relies on the watermarking of each individual queue. Two watermarks (high and low marks) are defined for each queue, depending on the parameters described below (section 3). The flow control operates as described on Figure 1:

1. free regime: starting from an empty queue, the connection is in its free regime, as long as the queue length does not reach the high mark  $SH$ . The rate allocated to this source by the upstream node is only constrained by the upstream node fair queuing algorithm (see below);
2. congested regime: when the queue reaches the high mark  $SH$ , a RM cell is sent upstream requesting the fair rate for this connection to be set at its minimum guaranteed rate,  $MCR_i$ , by the upstream node. The connection stays in the congested regime as long as its queue length stays above the low mark,  $SL$ . When the queue reaches the low mark  $SL$ , a RM cell is sent upstream allowing this connection to go back to its free regime.

The VP (Virtual Path) bandwidth is allocated to the active sources by a Weighted Round Robin mechanism, which guarantees that each active source is served at least at its minimum guaranteed rate. The bandwidth in excess of the sum of the minimum guaranteed rates is then allocated fairly among the active sources, according to their declared peak cell rates. The fair rates locally allocated to the sources depend on their free or congested state as notified by the downstream node:

- if the source is in the congested mode, its fair rate is its minimum guaranteed rate:  $R_i = MCR_i$
- if the source is in the free mode, its fair rate is given by

$$R_i = MCR_i + \frac{PCR_i}{\sum_{j*} PCR_{j*}} (VP - \sum_j MCR_j)$$

where  $\sum_{j*}$  means a summation on the free sources only.

Too large an excess bandwidth will lead to a low VP bandwidth utilization without any significant enhancement of the Quality of Service (QoS). Our purpose is to adapt periodically the VP bandwidth on each ATM node in order to maximize the VP bandwidth utilization without degrading the Quality of Service. The QoS indicator is chosen as the ratio of congested sources. This ratio is set to 20 %. The adaptation period is set to 5 seconds.

### 3 Definition of the state variables

As explained above, the neural network predictor requires a description of the state of the system ("state variables") to compute its prediction. In addition to the classical trade-off between the accuracy of the description and the complexity of the training, we had to take into account that



- the limited communication bandwidth left to collect the state variables in the interface cards and transmit them to the processing unit did not allow to collect state variables on a queue by queue basis with a sampling time compatible with our objectives of fast control of the VP bandwidth because of the possibly large number of VCs (up to 1024 VCs per VP);
- the limited processing power and storage space available on the interface cards did not allow to pre-process the raw queue data locally.

We had to rely on a sampling of the average state of the queues. Such a description requires a communication bandwidth independent of the number of VCs and does not require any processing power on the interface cards. The system is described by the ratio of the number of files found in a given state at sampling time to the total number of files:  $N_l$ , the ratio of files under the first mark,  $N_m$ , the ratio of files between the first and the second mark,  $N_h$ , the ratio of files upper the second mark and  $N_{cl}$ , the ratio of sources locally congested (i.e. sources for which a congestion notification has been sent to the upstream node).

The description of the system,  $D_t = (N_l(t), N_m(t), N_h(t), N_{cl}(t))$ , is transmitted to the processing unit at sampling time. Storage and pre-processing are done in the processing unit. Pre-processing consists of an averaging on jumping windows of size  $T$ :  $E_t = \frac{1}{T} \sum_{t-T+1}^t D_u$ . These averages are the state variables used by the neural network. In the following, the sampling period is 1 millisecond and the averaging period is 1 second.

## 4 A simple controller

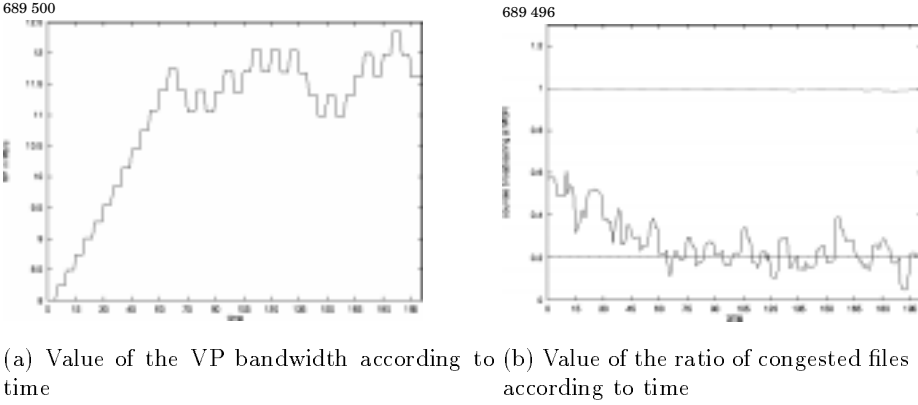
Since the Connection Admission Control (CAC) and the Usage Parameter Control (UPC) guarantee the conformity of a VC to its traffic contract, the connection blocking probability ( $N_{cl}$ ) can be increased without affecting the Quality of Service (QoS). Therefore, the method to control the VP bandwidth can be based on a connection blocking probability target ( $ON_{cl}$ ) [8]:

$$VP(t+1) = \begin{cases} (1+r)VP(t) & \text{if } N_{cl} > ON_{cl} \\ (1-r)VP(t) & \text{if } N_{cl} \leq ON_{cl} \end{cases}$$

The drawback of this step by step algorithm is the tuning of the parameter  $r$ . For a large value of  $r$ , the algorithm converges quickly with strong oscillations. For a small value of  $r$ , the convergence is slow. An improvement of this algorithm is to calculate  $r$  according to the value of  $N_{cl}$  [7]:  $r = \max(\gamma, \frac{N_{cl} - ON_{cl}}{1 - N_{cl}})$ , where  $\gamma$  is an a priori upper bound for the increment  $r$ .

The convergence of this algorithm is slow and it has only been proposed for adaptation periods of the order of the hour [8, 7].

Data traffic cannot be considered stationary on such a long period; indeed, the study of real traffic traces has shown that the local stationarity hypothesis holds on time scales of the order of a second only [9]. The adaptation of the VP bandwidth must cope with non-stationarities and therefore must converge on a time scale of a few seconds.



**Fig. 2.** Results of the step by step algorithm for  $ON_{cl} = 0.2$ .

## 5 A neural network controller

### 5.1 Learning a function

To control the VP bandwidth, a function using the state variables,  $E$ , as inputs and the VP bandwidth,  $VP$ , as output is needed (Figure 4). The regression consists to find a function  $f_\alpha \in \Gamma$  minimizing the functional risk  $R(f_\alpha) = \int (f_\alpha(E) - VP)^2 P(VP|E) dE$ .

In learning case, the conditional distribution of probability  $P(VP|E)$  is unknown and we approximate the functional risk by the empirical functional risk, using a set of examples  $R(f_\alpha) = \frac{1}{N} \sum_i^N (f_\alpha(E_i) - VP_i)^2$ , where  $N$  is the number of examples.

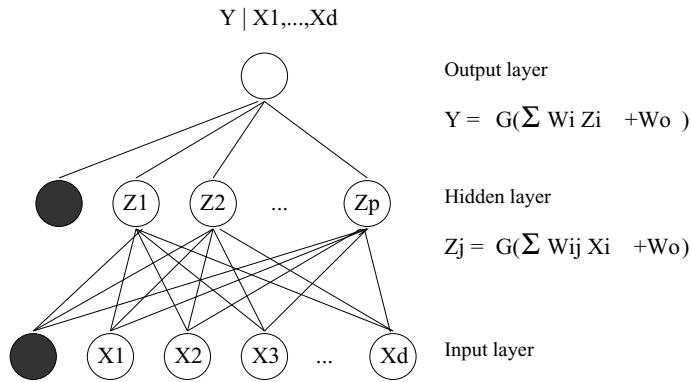
The learning machine defines the space of functions  $\Gamma$ . Neural networks (Figure 3) are powerful learning machines since they are non-linear models with efficient and robust training techniques.

### 5.2 Learning the optimal value of the VP bandwidth

Let  $F$  be the control function we want to learn:

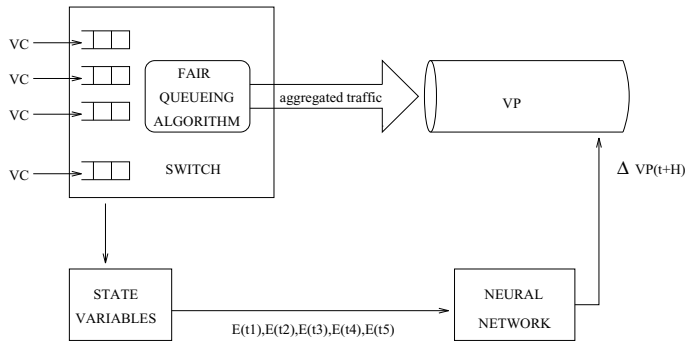
$$VP^*(t+H) = \max((1 + F(E_{t-4T}, E_{t-3T}, E_{t-2T}, E_{t-T}, E_t))VP(t), \sum_i MGR_i)$$

where  $H$  is the period of control ( $H = 5$  seconds in this study),  $E_t$  the averaged state variable at time  $t$ ,  $T$  an averaging period ( $T = 1$  second) and  $MGR_i$  the minimum guaranteed rate of the source  $i$ . To capture the temporal correlations, the inputs consists of a series of five consecutive values of the state variables:  $E_{t-4T}, E_{t-3T}, E_{t-2T}, E_{t-T}, E_t$ .



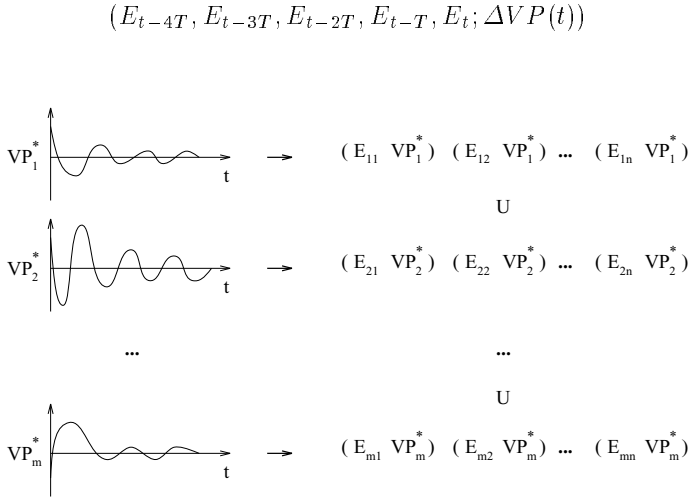
**Fig. 3.** The multilayer perceptron with one hidden layer allows to approximate non-linear functions.  $G(x) = \frac{1}{\exp(-kx) + 1}$  is the sigmoid function.

Note that the output of the controller cannot be lesser than the sum of minimum guaranteed rates so as to fulfill the traffic contract.



**Fig. 4.** Schema of the control process. The series of state variables  $E_t$  are the inputs of a neural network. The neural network returns the variation of the VP bandwidth needed to reach the optimal VP bandwidth for the next period  $H$ .

To obtain a fast control, we use the step by step algorithm previously described to generate data along a trajectory (Figure 2). Each trajectory generates several input examples with the same output: the optimal VP bandwidth (Figure 5). If such a function can be learned, it leads to a very fast convergence of the control process: anywhere on the trajectory, the function will return the optimal VP bandwidth. To obtain various outputs, we generate several trajectories, corresponding to different traffic contracts, source rates and initial VP bandwidths. The set of examples consists of 13000 examples. Each example consists of an input/output couple:



**Fig. 5.** The value of the optimal VP bandwidth is obtained at the end of the convergence of the step by step algorithm. For a trajectory  $i$ , at each control period  $j$  the state variables  $E_{ij}$  are collected to generate  $n$  input examples with the same output:  $VP^*$ . The use of  $m$  trajectories allows to generate  $N = mn$  different input/output couples.

We define the optimal value of the VP bandwidth,  $VP^*$ , according to the target congested rate  $ON_{cl}$ :  $VP^* = \{VP \text{ such that: } N_{cl} = ON_{cl}\}$ , as obtained at the end of the convergence of the step by step algorithm.

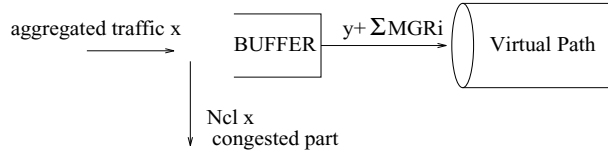
Finally, to minimize the functional risk between the learning machine and the target function  $F$ , we use:

- a multilayer perceptron as learning machine consisting of an input layer of twenty neurons corresponding to the series of state variables (five state variables, one state variable being a vector with four components), a hidden layer of five neurons and one output neuron,
- the standard back-propagation algorithm [6] to minimize the empirical functional risk on a training set of 10000 examples,
- a validation set to control the minimization of the functional risk: the learning is stopped when the empirical functional risk increases in the validation set composed by 3000 examples (examples which are not used to minimize the empirical functional risk).

## 6 A Kalman controller

To build the Kalman filter, we consider a simple model of the ATM node (Figure 6). The incoming traffic is aggregated. It is divided by the ATM node in two

parts: a part of the incoming traffic passing through the VP and a blocked and lost part. Under the assumption that all the sources send at least at their minimum guaranteed rate, we can write  $x = N_{cl}x + y + \sum_i MGR_i$ .



**Fig. 6.** The incoming aggregated traffic is divided in two parts. The first part,  $y$  and the sum of the minimum guaranteed rates, passes through the VP and the second part, the congested part  $N_{cl}x$  is blocked and lost.

Let  $V$  be the part of the VP bandwidth above the sum of the  $MGR$ :  $V = VP - \sum_i MGR_i$ .

We assume that the VP bandwidth utilization is high:  $y \approx V$ . Assuming Gaussian and centered noises, the following non-linear model defines the extended Kalman filter:

$$N_{cl} + \epsilon_{N_{cl}} = 1 - \frac{V}{x + \epsilon_x}$$

Where the incoming aggregated bandwidth  $x$  is the state variable, the excess bandwidth  $V$  the control variable, the ratio of congested sources  $N_{cl}$  the measured variable, and  $\epsilon_{N_{cl}}$  and  $\epsilon_x$  are respectively the measurement and the process noises. At each period  $H$ , the rate of congested sources  $N_{cl}(t)$  is measured, the noises  $(\epsilon_x, \epsilon_{N_{cl}})$  and the state variable  $x(t)$  are estimated by the Kalman filter. Then, the optimal VP bandwidth is given by:  $VP^*(t + H) = \hat{x}_t(1 - ON_{cl}) + \sum_i MGR_i$  where  $\hat{x}_t$  is the estimated state of the system by the Kalman filter and  $ON_{cl}$  the target congestion rate.

The initial value of  $\epsilon_{N_{cl}}$  is set according to the behaviour of  $N_{cl}$  after convergence of the step by step algorithm and the initial value of  $\epsilon_x$  is set according to the traffic contracts, assuming that the instantaneous rate of each source is drawn independently and uniformly between  $MGR_i$  and  $PCR_i$ .

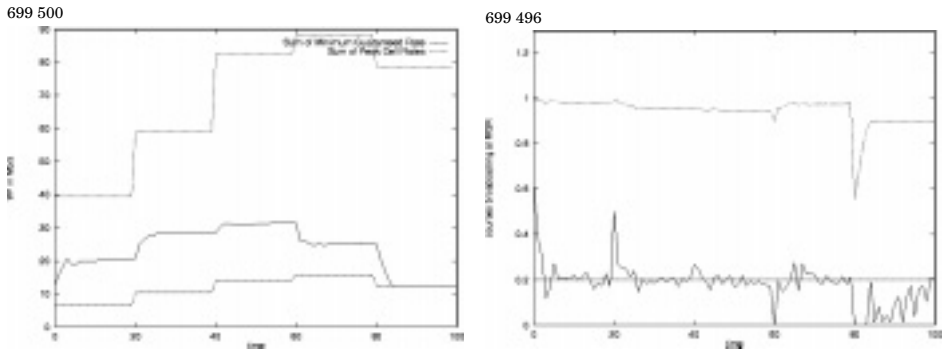
## 7 Experimental results

### 7.1 Simulation environment

Traffic and queue lengths measurements required functionalities which could not be implemented in the current version of the CMS switch and we had to rely on simulations. In order to measure the state variables, we had to simulate all the details of the implementation of the CMS switch as described above (per VC queuing, bandwidth sharing algorithm, flow control algorithm, see Section 2). A software-based simulator could not have allowed the implementation of our

training strategy in reasonable time; instead, we have developed a simulator of the CMS switch itself and mapped it on a hardware architecture.

With tools currently used in integrated circuit development such as hardware emulators, it is possible to reproduce a reliable image of the CMS switch by saving only useful functions for our implementation. Advantages of the hardware approach are [2] the simulation speed, close to the real switch, and the great number of VC that can be instantiated, leading to realistic traffic loads without slowing down simulation.



(a) Optimal VP bandwidth according to time. (b) Ratio of congested sources according to time, at the top the VP bandwidth utilization curve.

**Fig. 7.** Results of the neural network controller for  $ON_{cl} = 0.2$ . Every minute (20 time units) all the traffic contracts and rates of sources are randomly reset. On the last period, the optimal VP bandwidth is lower than the sum of minimum guaranteed rate.

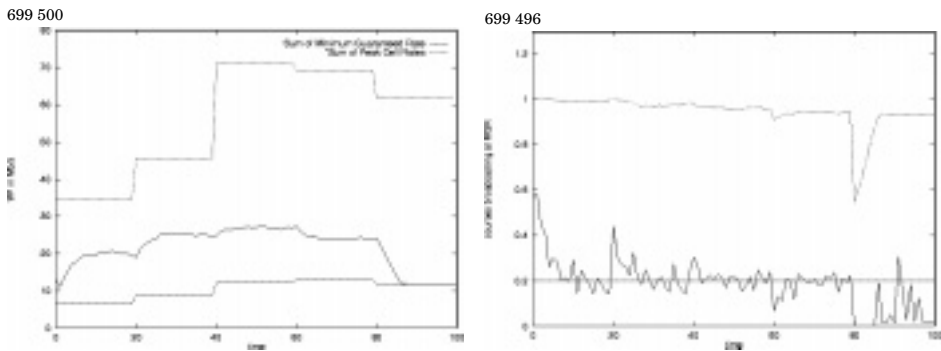
Indeed, with a reconfigurable hardware machine, speed is governed by the clock frequency from 1 to 10 MHz and this speed does not depend on the model complexity as for a pure software model. This allowed us to instantiate many sources simulating various VC. This hardware implementation can be achieved using usual tools in CAD design: a VHDL (VHSIC Hardware Description Language) description of the chip is used to describe the system in terms of concurrent processes, then a VHDL synthesiser translates the VHDL description into combinational logic and registers, the basic elements used in electronic systems, and finally, a compiler computes the link to map this hardware description on the sea of gates of the emulator. The use of an emulator is original in this approach. This machine is composed of a sea of gates that can be dynamically linked to reproduce an electronic design. The M500 emulator from Metasystems comprises 500,000 programmable logic gates, connectable to each other, 17 Mbytes of memory, single or double port, an adjustable clock frequency from 1 to 10 MHz. The M500 emulator and the associated software has important features for debugging: it stores signal values of the last 7000 clock cycles, allows

the user to start, stop, and run clock step by clock step simulation as desired. Reconfigurable hardware machines are therefore very appealing simulation tools for complex systems such as the dynamical coupling between the bandwidth sharing and flow control as implemented in the CMS node.

The simulator comprises three main blocks [1]:

- Traffic sources, most widely used statistical models are implemented: periodic sources with a period modified regularly, uniform traffic and geometric process, Markov modulated Bernoulli process with on-off sources, which simulated bursty traffic.
- Switch model: the CMS model can receive up to 64 sources or VC to fit in the hardware emulator.
- Traffic analysis: several quantities (such as cell number, lost cell number, inter-arrival process, queue occupancy, dates when SL and SH marks are reached) can be stored “on the fly” to evaluate the state variables.

This simulator has been used for the generation of the training and validation sets and for the test experiments presented below.



(a) Value of the VP bandwidth according to time. (b) Value of the ratio of congested files according to time, at the top the VP bandwidth utilization curve.

**Fig. 8.** Results of the Kalman controller for  $ON_{cl} = 0.2$ . Every minute (20 time units) all the traffic contracts and rates of sources are randomly reset.

## 7.2 Discussion

Corresponding to the implementation constraints, the prediction period  $H$  is 5 seconds the sampling period of the state variables is 1 ms and the target ratio of congested sources is  $ON_{cl} = 0.2$ . To evaluate and to compare the control of the Kalman filter and of the neural network filter, we generated 64 various periodic

traffics on a period of one minute. After this minute, we introduced an important non-stationarity by randomly resetting the traffic contracts and rates of all the sources. The emission rates and traffic contracts of the sources were drawn from Gaussian distributions (with the constraint that the emission rate cannot exceed the declared peak cell rate; note that sources can emit at a lower rate than their minimum guaranteed rate). We introduced a increasing trend of the means of Gaussians on a first period and a decreasing trend on a second period.

The neural network experiment and the Kalman filter experiment were built using the same methods but with slightly different values of the mean of Gaussians generating the parameters of sources. The quality and stability of the convergence can be compared on the stationarity period and the speed of convergence on the non-stationarity period.

First, the purpose is to maximize the VP bandwidth utilization. In both cases, it is close to 1 (Figure 7b, 8b). The analysis of the curves shows that the control of the Kalman filter (Figure 8a) is slower than the one of neural network (Figure 7a): a stable value of the VP bandwidth is reached on one or two iterations (5 to 10 seconds) in the case of neural network and two or three iterations for the Kalman filter. Moreover, the quality and stability of the neural network controller (Figure 7b) is better than the Kalman controller (Figure 8b): the oscillations of the measured congested rate have lower amplitudes for the neural network controller than for the Kalman controller.

It may be argued that the switch model designed for the Kalman filter is grossly simplified as it implements almost nothing of the complex queue management, fair service and congestion control described above in Section 2. The problem is that modeling such a complex system in greater details leads very fast to untractable models.

This illustrates the difference between the two kinds of controllers when dealing with complex systems: the "mapping approach" (neural network controller in this work) allows us to describe the system in great details (the hardware simulator implements all the complexities of the CMS switch, so that the state variables describe the behaviour of true switch), to the expense of finding a good mapping between high dimensional spaces by an off-line training, whereas the "modeling approach" (Kalman controller in this work) forces us to use simplified models but allows us to adapt the model parameters on-line.

Training of the neural network is performed off-line and gives superior performances, but it must be emphasized that the performances will stay good only as long as the examples used for the off-line training will be representative of the behaviour of the system. This implicit stationarity hypothesis is certainly wrong in the long term and appropriate strategies (such as periodic re-training) have to be developped to cope with non-stationarities.

The modeling approach has good although slightly inferior performances but these performances will stay good as long as the model is representative of the behaviour of the system and non-stationarities are coped with by the adaptivity of the Kalman filter.



## 8 Conclusion

We have described, implemented and compared the performances of two controllers of very different designs (a Kalman filter and a neural network) so as to adaptively dimension the VP bandwidth between two nodes so as to keep the quality of service close to a target value.

Simulations were made possible by the use of a hardware emulator allowing to reproduce faithfully the complex behaviour of the switch.

Both controllers showed good performances with a faster and more accurate control by the neural network. We have discussed how this improved performance came at the expense of adaptivity to long-term non-stationarities.

Both controllers are currently being implemented in a prototype of the CMS switch so as to test their performances under real workload.

## References

1. C. Labbé, V. Olive, and J.M. Vincent. Emulation on a versatile architecture for discrete time queuing network : Application to high speed network. In *ICT98, International Conference on Telecommunications*, 1998.
2. C. Labbé, F. Reblewski, and J.M. Vincent. Performance evaluation of high speed network protocols by emulation on a versatile architecture. *Recherche Opérationnelle/Operations Research*, 32(3), 1998.
3. I. Norros. On the use of fractional brownian motion in the theory of connectionless networks. *IEEE Journal on Selected Areas in Communications*, 13(6), 1995.
4. V. Paxson and S. Floyd. Wide area traffic: the failure of poisson modelling. In *Sigcomm'94, Computer Communication Review*, volume 24, pages 257–268, 1994.
5. R. Riedi and Willinger W. Towards an improved understanding of network traffic dynamics. In *Self-similar Network Traffic and Performance Evaluation*, Wiley, 1999.
6. D. E. Rumelhart, G. E. Hinton, and R. J. Williams. Learning internal representations by error propagation. In *Parallel Distributed Processing: Explorations in the Microstructures of Cognition*, volume 1, 1986.
7. C. Shioda, H. Toyozumi, H. Yokoi, T. Tsuchiya, and H. Saito. Self-sizing network: a new network concept based on autonomous vp bandwidth adjustment. In *ITC 15*, 1997.
8. S. Shioda. Evaluating the performance of virtual path bandwidth control in atm networks. *IEICE Trans. Commun.*, E77-B(10):1175–1187, 1994.
9. S. Vaton, E. Moulines, and H. Korezlioglu. Statistical identification of lan traffic data. In *5th IFIP Workshop on Performance Modelling Evaluation of ATM Networks*, 1997.
10. W. Willinger, V. Paxson, and M.S. Taqqu. "A Practical Guide to Heavy Tails : Statistical Techniques for Analysing Heavy Tailed Distributions. Birkhauser Verlag Boston, 1998.

# Fairness and Aggregation: A Primal Decomposition Study \*

André Girard<sup>1</sup>, Catherine Rosenberg<sup>2</sup>, and Mohammed Khemiri<sup>3</sup>

<sup>1</sup> INRS-Télécommunications Place Bonaventure, 900 de la Gauchetière Ouest, Niveau C, C.P. 644, Montréal (Qué) Canada H5A 1C6 [andre@inrs-telecom.quebec.ca](mailto:andre@inrs-telecom.quebec.ca)

<sup>2</sup> Department of Electrical and Computer Engineering, 1285 Electrical Engineering Building, Purdue University, West Lafayette IN 47907 - 1285 USA  
[cath@ecn.purdue.edu](mailto:cath@ecn.purdue.edu)

<sup>3</sup> Ecole Polytechnique, Paris, France

**Abstract.** We examine the fair allocation of capacity to a large population of best-effort connections in a typical multiple access communication system supporting some bandwidth on demand processes. Because of stringent limitations on the signalling overhead and time available to transmit and process information, it is not possible to solve the allocation globally to obtain the optimally fair allocation vector. A two-level procedure is proposed where connections are aggregated within terminals, which send aggregated requests to the controller. The controller then computes the appropriate aggregated allocation per terminal and sends the corresponding allocation vector back to the terminals. Each terminal then computes the allocation vector for its connections. We want to study what aggregated information the terminals should send and what allocation problem should the controller and the terminals solve to produce a near-optimal (in terms of fairness) allocation vector. We propose a primal-based decomposition approach, examine in detail a number of approximations and show that by transmitting its total demand and number of connections, each terminal can achieve a near-optimal and near-fair allocation of capacity.

## 1 Introduction

The fair allocation of capacity among best-effort users is a subject that is gaining wide interest both in wire-line networks, such as ABR services in ATM [1, 2], non-QoS constrained services in IP networks [3] and in wireless and wireline access and satellite systems. In this paper, we focus on systems with multiple access links for which a process is needed to share the available capacity on a demand basis. All these systems are characterized by a set of users, each submitting a request for bandwidth, the sum of which usually exceeds the available total bandwidth. The required bandwidths are not absolutely needed and the users are willing to live with an allocation smaller than what they requested. The

---

\* This work was performed while the two last authors were at Nortel Networks, Harlow, UK.

problem is then to allocate the total available bandwidth in an optimally fair manner. In all that follows, the measure of optimality always refer to the fairness of the allocation process unless otherwise noted.

Although the fair allocation problem is relatively simple to solve to optimality, in practice, there are a number of situations where the actual computation turns out to be difficult. Consider for instance the case of a large population of best-effort connections on a satellite system having typically many thousand terminals, each with potentially several connections. Because the bandwidth of the multiple access uplink is very limited, there is a need to limit the signalling overhead, i.e., the amount of information sent to request bandwidth. Also, the bandwidth allocation process is performed periodically every few tens of milliseconds for geo-stationary satellite systems and the allocation process should be able to run within this time limit. One of the main issues is that the problem of sharing fairly a given capacity among a large number of un-coordinated users competing for this capacity is very time consuming and there will be a scalability issue if we want to design a fast, low-overhead and optimally fair allocation process.

These systems, however, have a hierarchical structure where a terminal will manage a number of connections. These terminals will have some processing power and are only able both to request bandwidth from the controller and to reallocate this bandwidth among their connections. With a structure like this, each terminal receiving (or computing) the requests from its own connections could transmit to the controller some aggregated information based on the individual requests. In such a system, two questions naturally come to mind and form the subject of this paper: 1) How should the individual call requests be aggregated by the terminals, i.e., what aggregated information should each terminal send on behalf of its connections? and 2) how good are these approximations with respect to the optimal allocation?

The main findings of this work is that if each terminal sends only the sum of the requests of each of its connections, the controller cannot allocate the available capacity in a near optimal fashion. We can get a much better solution if each terminal sends both the sum of the requests of each of its connections and the number of connections. Based on these requests, the controller can solve an optimization problem of lower complexity (as opposed to the one it would have to solve if each connection had sent its own request) and thus is able to perform the computation within the allowed time. We show that this process yields results very close to the optimal solution.

The paper is structured as follows. First, we state the model and propose a primal decomposition of the optimal allocation. We solve the sub-problems and state the master problem. Then, we investigate two types of approximations. The first set uses only the sum of the individual requests for each terminal while in the second set, each terminal is allowed to send an aggregated request made up of two terms, one of which being the above sum. Numerical results then show that sending the number of connections as the second term yields a nearly optimal solution and that this is probably the best trade-off that can be done

in terms of efficiency and signalling overhead. Conclusions and future extensions to the work follow.

## 2 Optimal Allocation Model

We want to compute the optimally fair allocation vector  $\mathbf{x}$  by solving the problem

$$\max_{\mathbf{x}} Z = \prod_{i=1}^n x_i \quad (1)$$

$$\sum_{i=1}^n x_i = C \quad (2)$$

$$0 \leq x_i \leq d_i \quad (3)$$

where  $n$  is the total number of connections,  $C$  is the total available capacity,  $x_i$  is the allocation to connection  $i$  and  $d_i$  is the request made by connection  $i$ . Note that there may be more constraints on the terminals but that would not fundamentally impact the results of this study. The problem is interesting when  $\sum_{i=1}^n d_i > C$  that is, when it is not possible to meet all the requests. In a satellite system,  $n$  will be of the order of several thousands and the problem has to be solved once every few tens of milli-seconds. At each period, the available capacity  $C$  will change due to the arrivals and releases of calls that are not best-effort (i.e., that are allocated some resource on a reservation or static basis) so that we do not expect the problems to be very similar from one period to the next. Hence proposals based on explicit knowledge of the demand for each connection in the system are not realistic in terms of signalling overhead as well as processing time and power.

## 3 A Primal Decomposition Method

The basic idea for reducing the complexity of the computation is to use a two-level allocation procedure. At the terminal level, the controller partitions the total capacity  $C$  among the terminals according to some rule, yielding an allocation  $C_i$  to terminal  $i$ . Once this allocation is made, each terminal allocates its  $C_i$  among its connections.

An important advantage of a primal decomposition method is that the solutions are always feasible with respect to the total capacity constraint (2). This is in sharp distinction with dual methods where this constraint is not met unless the multiplier has been exactly calculated. In the present case, because there is little time for iterations, it is expected that dual methods would not have time to converge and thus could yield poor solutions and this is why we concentrate on primal techniques.

### 3.1 The Primal Decomposition Model

We now give a precise definition of the primal decomposition model for problem (1–3). Suppose that we have allocated a capacity  $C_i$  to terminal  $i$  by some yet unspecified method. Once this allocation has been made, assume also that for each terminal  $i$ , we allocate  $C_i$  optimally among the  $n_i$  connections of this terminal. Let  $d_{i,j}$  and  $x_{i,j}(C_i)$  be the demand and the optimal capacity allocated to connection  $j$  of terminal  $i$  and define the value of the terminal  $i$  objective function as

$$P_i(C_i) = \prod_{j=1}^{n_i} x_{i,j}(C_i).$$

We write  $P_i(C_i)$  and  $x_{i,j}(C_i)$  because once the terminal allocation  $C_i$  is known, the allocation to the connections and the value of the objective function are completely defined. The optimal allocation problem can then be written as

$$\max_{C_i \geq 0} P = \prod_{i=1}^m P_i(C_i) \text{ subject to } \sum_{i=1}^m C_i \leq C \text{ and } C_i \leq D_i \quad (4)$$

where  $m$  is the total number of terminals and  $D_i = \sum_j d_{i,j}$  is the sum of the requests for all the connection in terminal  $i$ . Each of the  $P_i(C_i)$  is computed solving the following optimal allocation for terminal  $i$

$$\max_{x_j} P_i(C_i) = \prod_{j=1}^{n_i} x_j \text{ subject to } \sum_{j=1}^{n_i} x_j \leq C_i \text{ and } 0 \leq x_j \leq d_j \quad (5)$$

where  $n_i$  is the number of connections of terminal  $i$  and vector  $x_j$  is the allocation vector for terminal  $i$  where we have dropped the terminal index  $i$  for simplicity. Note that problem (4) is entirely equivalent to problem (1–3) under the condition that the  $P_i(C_i)$ s are calculated by solving (5).

### 3.2 Solving the Sub-Problems

We examine the structure of the sub-problem (5) and propose a fast solution technique. Note that this problem has exactly the same structure as the original problem (1–3) but is of much smaller size since it deals only with the connections of a particular terminal  $i$ .

It is obvious that the objective function of (5) is monotone increasing in  $x_j$ . The maximum will most likely be on one of the vertices of the domain and the first order optimality conditions are not very useful for computing a solution. The hard part is rather to determine on *which* vertex lies the optimal solution.

Assume that for a given value of  $C_i$ , we have chosen a set  $J_i(C_i)$  of constraints of the form  $x_j \leq d_j$  to be saturated and let  $k_i(C_i)$  be the number of such constraints. The optimal solution of the sub-problem for the given value  $C_i$  then becomes

$$x_j \begin{cases} = d_j & \text{if } j \in J_i(C_i) \\ < d_j & \text{otherwise.} \end{cases}$$

The choice of  $J_i(C_i)$  is of course subject to the condition  $\sum_j x_j \leq C_i$ . If this condition is not met, we must choose another set of saturated constraints. We must then compute the values of the  $x_j, j \notin J_i(C_i)$ . We rewrite the problem (5) for this particular choice of saturated constraints

$$P(J_i) = \max \prod_{j \notin J_i(C_i)} x_j \text{ subject to } \sum_{j \notin J_i(C_i)} x_j = \bar{C}_i$$

where  $\bar{C}_i = C_i - \sum_{j \in J_i(C_i)} d_j$  is the residual capacity not allocated to the saturated constraints and we write  $P(J_i)$  to indicate that the value of the objective depends on the choice of the saturated constraints. Note also that since all the saturated constraints are in  $J$ , there are no bounds on the remaining variables. We can then use the first order optimality conditions to compute these variables and we find the optimal solution

$$x_j = \begin{cases} d_j & \text{if } j \in J_i(C_i) \\ \frac{\bar{C}_i}{n_i - k_i(C_i)} & \text{otherwise,} \end{cases}$$

in other words, the residual capacity is allocated equally among the unsaturated connections. An implicit condition is that this allocation does not exceed the bounds  $d_j, j \notin J_i(C_i)$ . If this is not the case, then the choice of  $J_i(C_i)$  has to be modified. The optimal value of the objective function of the sub-problem can then be written

$$P(J_i) = \prod_{j \in J_i(C_i)} d_j \left[ \frac{C_i - \sum_{j \in J_i(C_i)} d_j}{n_i - k_i(C_i)} \right]^{n_i - k_i(C_i)}. \quad (6)$$

The problem then reduces to choosing the set  $J_i$  that will give the largest value for  $P(J_i)$ . We can prove the following theorem (the proof is omitted because of lack of place):

**Theorem 1.** *The optimal solution of the sub-problem, for a given value of  $C_i$ , is obtained by saturating the largest possible number of constraints with the smallest possible values for the  $d_j$ s.*

Once we have made this choice, the residual capacity is spread equally among the unsaturated connections.

We can propose a very simple algorithm for this particular allocation. Imagine that  $C$  is increasing from 0. As long as the smallest value of the  $d_j$ s is not reached, we allocate  $C$  equally to all connections. When the smallest bound is reached, the corresponding value of  $x_j$  is set at this bound and we keep allocating the remaining capacity among the still unsaturated connections. This goes on until  $C$  has reached its real value.

### 3.3 Solving the Terminal Allocation Problem

Given the structure of the sub-problems and their optimal solutions, we now turn to the solution of the terminal allocation problem (4). From now on, we assume that the  $d_j$ s are numbered by increasing value. The controller can calculate an optimal solution if it knows the value of the functions  $P_i(C_i)$  for all terminals and any value  $C_i$ , which means that the computation and transmission times will be large. We now describe the form of these functions.

Using the logarithmic form of the problem, we write eq. (6) as

$$\log P_i(C_i) = \sum_{j \in J_i(C_i)} \log d_j + [n_i - k_i(C_i)] \{ \log \bar{C}_i - \log [n_i - k_i(C_i)] \}.$$

We know that  $k_i(C_i)$  is a monotone increasing function of  $C_i$ . Assume that  $C_i \approx 0$ . In that case, no constraint can be saturated and we have  $J_i(C_i) = \emptyset$ ,  $k_i(C_i) = 0$  and

$$P_i(C_i) = \left( \frac{C_i}{n_i} \right)^{n_i}. \quad (7)$$

In other words, near 0,  $\log P_i(C_i)$  is linear in  $\log C_i$  with a positive slope  $n_i$ . This situation remains as long as  $C_i$  is not large enough to allocate a capacity of  $d_1$  to all connections, that is, as long as  $C_i \leq n_i d_1$ . At that point, the first constraint  $x_1 \leq d_1$  becomes saturated. If we keep increasing  $C_i$ , we get  $\log P_i = \log d_1 + (n_i - 1) [\log(C_i - d_1) - \log(n_i - 1)]$  and this as long as  $n_i d_1 < C_i \leq d_1 + (n_i - 1)d_2$ . In that range,  $\log P_i$  is a linear function of  $\log(C_i - d_1)$  with a positive slope of  $n_i - 1$ . When  $C_i \rightarrow \infty$ , we can allocate the full demands to all connections and we have

$$\log P_i(C_i \approx \infty) = \sum_j \log d_j$$

and this is independent of  $C_i$  so that we call this the *uniform* approximation. We can write the general form for  $\log P_i(C_i)$  if we first consider the points  $0, n_i d_1, d_1 + (n_i - 1)d_2, \dots, \sum_{l=1}^k d_l + (n_i - k)d_k \dots \sum_{l=1}^{n_i} d_l$  on the  $C_i$  axis. These points define a set of intervals  $\mathcal{I}_k, k = 0 \dots n_i - 1$  where the function takes a different form in each interval. More precisely, we have, for  $k = 0 \dots n_i - 1$ , when  $C_i \in \mathcal{I}_k$ ,

$$\log P_i(C_i) = \sum_{j=1}^k \log d_j + (n_i - k) \left[ \log \left( C_i - \sum_{j=1}^k d_j \right) - \log(n_i - k) \right] \quad (8)$$

where a sum with its upper limit lower than its lower one is defined as 0. The function  $\log P_i(C_i)$  is made up of logarithmic segments and is continuous and monotone increasing. We can then solve problem (4) with (8) as the objective function.

There are two difficulties with this exact model. First, given the form of the functions (8), the solution of the main allocation problem (4) is likely to be difficult since the objective function is nonlinear and not differentiable. The second

and more serious difficulty is that the exact calculation of the functions (8) requires the knowledge of *all* the requests for *each* connection of all the terminals (i.e., all the  $d_{i,j}$ ). This is not really surprising since we insist on calculating a truly optimal solution and we should expect that we would need as much information as if we were solving directly problem (1–3). We know that this is not feasible in practice and we want to investigate methods that would require less information and also perhaps make for an easier allocation problem.

## 4 Approximations

We examine here a number of approximations to the exact allocation problem in order to limit the information to be sent by the terminals to the controller. We proceed as follows. We determine what information is available and we describe a simple allocation procedure that has already been suggested elsewhere or that seems natural. We then translate this procedure into an approximation of the  $P_i(C_i)$  functions. In this way, we can immediately see whether an approximation has a chance of being reasonably accurate or not. In all cases, we assume that the value of  $D_i$  is available for all terminals. We first examine a simple allocation procedure based only on the knowledge of the  $D_i$ s and then we extend the analysis to cases where another set of numbers is available, either the number of connections per terminal (the  $n_i$ s) or the product of the requests  $\prod_j d_{i,j}$ .

### 4.1 Approximation with $D_i$ only

The simplest scalable technique is a two-level algorithm based on summing the demands in each terminal and sending them to the controller that allocates fairly and optimally the available capacity among the terminals based on these aggregated demands. Once the terminal allocation is done and received by the terminals, each terminal allocates fairly its quota among its connections. The  $C_i$ s are calculated by solving the optimal terminal allocation problem

$$\max_{C_i \geq 0} P = \prod_{i=1}^m C_i \text{ subject to } \sum_{i=1}^m C_i = C \text{ and } C_i \leq D_i. \quad (9)$$

This amounts to replacing the exact  $P_i(C_i)$  function of eq. (4) by the linear approximation  $P_i(C_i) = C_i$ . Going back to eq. (7), we see that this is equivalent to assuming  $n_i = 1$ . This is hardly surprising since we have a single value to characterize terminal  $i$ .

Once this solution is obtained, each terminal  $i$  allocates its  $C_i$ s by solving the sub-problem (5). We can also see on Fig. 1 the plot of approximation (9) with the exact solution and the upper bound. We have also tried another method in which each terminal computes its own estimate of a best value for the number of connections and then uses this to do the allocation. In that case, the terminal allocation problem becomes degenerate and numerical results have shown that this yields to a very poor approximation of  $\log P_i(C_i)$ . For these reasons, this



approach has not been pursued. It should be quite clear that the more or less obvious approximations involving only the values of  $D_i$  are not going to give very good results and that more information is needed.

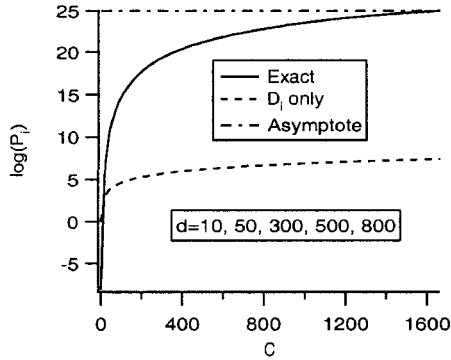


Fig. 1. Comparison of uniform approximation with the exact solution

#### 4.2 Approximations with $D_i$ and $n_i$

Given that the solutions computed from the models with only the  $D_i$ s available can be relatively poor, we suggest that more information about the terminals should be used in the solution of the terminal allocation problem. In order to minimize the signalling overhead, we use only one additional value per terminal and the one that readily comes to mind is the  $n_i$ s, the number of connections in terminal  $i$ . Hence each terminal  $i$  sends an aggregated request made up of two terms:  $D_i$  and  $n_i$ .

We consider first the simplest allocation technique that makes use of both the  $D_i$ s and the  $n_i$ s. Let  $n = \sum_i n_i$  be the total number of connections. A straightforward method would be to allocate  $C/n$  to each connection, which assumes in effect that all connections have the same request. For this reason, we call this the *uniform* approximation. Each terminal that has a total request  $D_i \leq C_i = n_i(C/n)$  gets its allocation  $D_i$ . This is subtracted from  $C$ , leaving a residual capacity  $C'$ . This in turn is allocated among the remaining terminals with the same rule, and so on until all the capacity has been allocated. Once the allocation  $C_i$  has been computed, each terminal solves its own allocation problem (5) as before.

The question is then what terminal allocation problem is being solved by this algorithm. Let  $T_i$  be the set of connections that belong to terminal  $i$ . Rewrite problem (1-3) as:

$$\max Z = \prod_i \prod_j x_{i,j} \text{ subject to } x_{i,j} \leq d_{i,j} \text{ and } \sum_{i,j} x_{i,j} = C. \quad (10)$$

The uniform allocation can be rewritten as

$$x_{i,j} = y_i \quad \forall j \in \mathcal{T}_i \text{ and } d_{i,j} = \frac{C}{n} \quad \forall (i, j)$$

$$\max Z = \prod_i (y_i)^{n_i} \text{ subject to } y_i \leq \frac{C}{n} \text{ and } \sum_i n_i y_i = C$$

and if we define  $X_i = n_i y_i$ , we get the equivalent form

$$\max Z = \prod_i \left( \frac{X_i}{n_i} \right)^{n_i} \text{ subject to } X_i \leq n_i \left( \frac{C}{n} \right) \text{ and } \sum_i X_i = C \quad (11)$$

and we see that in that case, we get the power allocation

$$P_i(C_i) = \left( \frac{C_i}{n_i} \right)^{n_i}. \quad (12)$$

This is the first segment of the real function as can be seen from Eq. (7).

### 4.3 Approximation with $D_i$ and $A_i$

Although the  $n_i$  are an obvious choice to transmit in addition to the  $D_i$ s, this is not the only possibility. Another quantity that is readily available is the product of the demands  $A_i = \prod_{j=1}^{n_i} d_{i,j}$  which could be used to obtain a better allocation of the terminal capacities.

One way to use the value of  $A_i$  would be to assume that there are two connections for each terminal and to try to determine their requests  $\bar{d}_1$  and  $\bar{d}_2$  such that  $\bar{d}_1 + \bar{d}_2 = D$  and  $\log(\bar{d}_1 \bar{d}_2) = \log(A)$ . Unfortunately, it is not difficult to find examples of request vectors  $d_j, j = 1 \dots n$  such that there do not exist two real values for  $\bar{d}_1$  and  $\bar{d}_2$ .

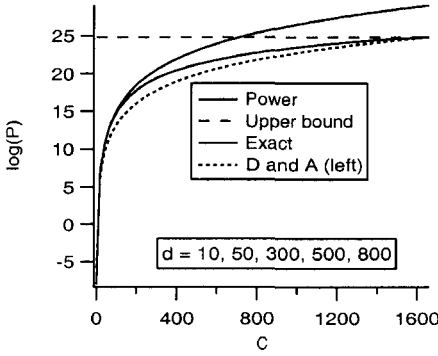
Although approximation (12) seems to be reasonably accurate, we can see that it overestimates the exact solution at the boundary  $C_i = D_i$ . Letting  $\tilde{P}_i(C_i)$  denote the value of approximation (12), we get at the boundary

$$\tilde{P}_i(D_i) = \left( \frac{D_i}{n_i} \right)^{n_i} \text{ and } P_i(D_i) = \prod_j d_{i,j}$$

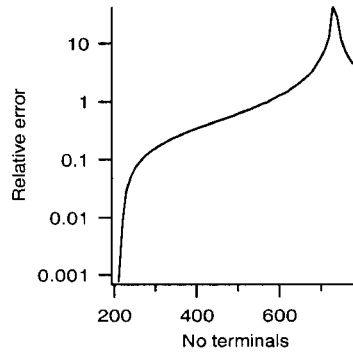
which are different. We could hope for a better fit if we could use an approximation of the form (12) but with the requirement that it should go through the exact value of  $P_i(D_i)$  at  $C_i = D_i$ . In order to do this, we need the value of  $\log A_i = \sum_j \log d_j$ . We want the terminal to determine and send an  $n^*$  such that

$$\log A = n^*(\log D - \log n^*). \quad (13)$$

This function is concave and can have 0, 1 or 2 solutions depending on whether  $D/e$  is larger (2 solutions), equal (1 solution) or smaller (no solution)



**Fig. 2.** Comparison of three approximations



**Fig. 3.** Relative error with the uniform approximation and the  $D_i$ s only

than  $\log A$ . In practice, it seems that there are generally two solutions and that the smaller one is much better than the larger one. A comparison of the four approximations is shown on Figure 2. The case corresponding to the larger solution of eq. (13) is not shown because it is not very good. We can see that the uniform approximation is an overestimate of the true function while the use of the product of the demands yields a lower bound.

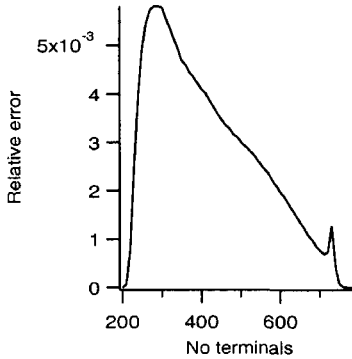
## 5 Accuracy of the Approximations

The accuracy of the approximations can be checked by comparing with the optimal solution of the total problem. The values used to plot the figures are computed with 1000 random cases for each given number of terminals. Each case is obtained by generating an independent random value of  $n_i$  (number of connections per terminal  $i$ ) and  $d_{i,j}$  (the request of each connection  $j$  of terminal  $i$ ) with  $P(n_i = k) = 0.09$  for  $1 \leq k \leq 10$  and equally distributed for the remaining values up to a value of 32 and  $d_{i,j}$  equally distributed between 1 and 9. These connections are competing for an overall capacity equal to 5120 units corresponding to 32 time slots and 160 carriers in a (MF-TDMA) Multi Frequency Time Division Multiple Access uplink.

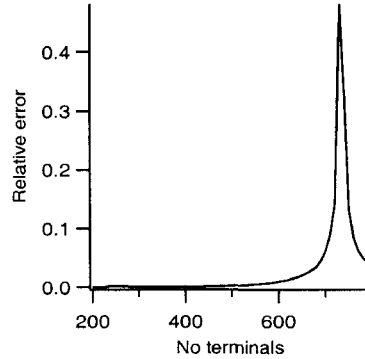
A first criterion that we use is the relative error  $\Delta$  defined as  $\Delta = |Z - Z^*|/|Z^*|$  where  $Z$  is the value of the objective function obtained with the approximation and  $Z^*$  is the optimal value of the objective function.

We can already see that the approximations of the  $P_i(C_i)$  functions with  $D_i$  only have a very poor accuracy and this can be checked in Fig 3 where we show on a logarithmic scale the relative error as a function of the number of terminals. We can see that this relative error varies widely, in some cases being more than a factor 10. In other words, for some configurations, the fairness of the approximate allocation is more than 10 times poorer than the optimal allocation. The really interesting cases are the ones with an aggregate request based on two

terms, the  $D_i$ s and either the  $n_i$ s or the  $A_i$ s. The first case is compared in Fig 4 and the second in Fig. 5 where we have plotted the relative error as a function of the number of terminals. In both cases, when there is a small number of terminals  $m$ , all demands can be met and all methods give identical results, which explains why the curves go to zero near the origin. Similarly, when  $m$  is large, it is impossible to satisfy any request and in that case also all the methods are equivalent. In the middle region, the peak of the error curve occurs when the approximate values coincide with  $Z^*$  and in this region, the approximation where we send the values of the  $n_i$ s is much better than the one where we use the  $A_i$ s.



**Fig. 4.** Relative error of the approximation with the  $D_i$ s and  $n_i$ s

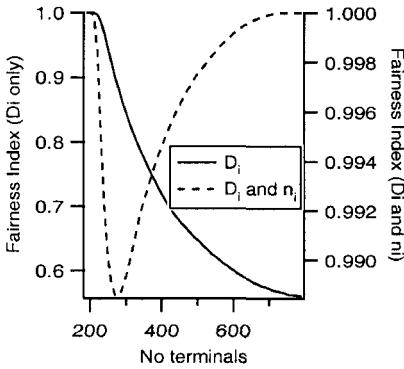


**Fig. 5.** Relative error of the approximation with the  $D_i$ s and  $A_i$ s

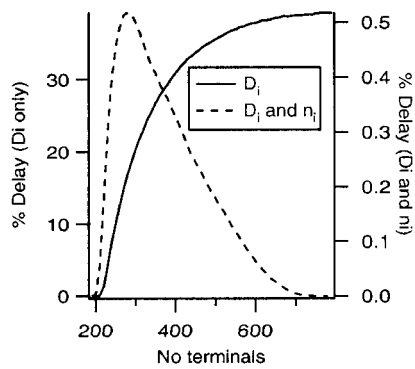
Another criterion to quantify numerically the unfairness of a scheme is the fairness index introduced by K.Jain in [4]. Suppose that an approximate solution allocates  $\bar{x}_1, \bar{x}_2, \dots, \bar{x}_n$  instead of the optimal allocation  $x_1^*, x_2^*, \dots, x_n^*$ . Defining for each connection  $j$  the normalized allocations  $x_j = \bar{x}_j / x_j^*$ , the fairness index  $\gamma$  is then defined as  $\gamma = \left( \sum_j x_j \right)^2 / \left( n \sum_j x_j^2 \right)$

If the allocation given by the approximate solution is the same as the optimal one, then  $\gamma = 1$ , and the system is 100% fair. As the disparity increases, fairness decreases and a scheme which favors only a selected few connections has a fairness index near 0. The results shown on Fig. 6 indicate that for the approximation with  $D_i$  only, the scheme can be only 55% fair. In other words, this scheme favors 55% of the connections and discriminates the others. However, the scheme using  $D$  and  $n$  is almost 99% fair.

A third criterion has been proposed by L.Massoulié in [5] for file transfers. In that case, a legitimate bandwidth sharing objective would be to minimize the time needed to complete the transfers. This is done via a potential delay equal to the reciprocal of the rate allocation  $1/x_j$  and by finding the allocations that minimizes the total potential delay  $\sum_j 1/x_j$ . In fact, we can show that the



**Fig. 6.** Fairness measure of the approximations with  $D_i$  only and  $D_i$  and  $n_i$



**Fig. 7.** Relative increase in potential delay

algorithm for the main problem (1–3) also minimizes the total potential delay. We can see on Fig. 7 that the approximation using  $D_i$  increases the optimal potential delay by 40% whereas its increase is less than 0.5% for the approximation with  $D$  and  $n$ .

The conclusion is quite clear that the best accuracy is obtained when each terminal transmits both  $D_i$  and  $n_i$ . Given the high accuracy of the approximation, we feel that it is not necessary to look for better approximations and that a two-level algorithm with these parameters is sufficient.

## References

1. H. Yaiche, R. Mazumdar, and C. Rosenberg, "A game theoretic framework for rate allocation and charging of available bit rate (ABR) connections in ATM networks," in *Proc. Broadband'98*, P. Kühn, Ed., 1998.
2. A. Arulambalam and X.Q. Chen, "Allocating fair rates for available bit rate service in ATM networks," *IEEE Communications Magazine*, pp. 92–100, 1992.
3. A. Charny, D. Clark, and R. Jain, "Congestion control with explicit rate indication," in *Proceedings of the IEEE International Conference on Communications*, June 1995, IEEE.
4. R. Jain, D. Chiu, and W. Hawe, "A quantitative measure of fairness and discrimination for resource allocation in shared systems," Tech. Rep. DEC TR-301, Digital Equipment Corp., Sept. 1984.
5. L. Massoulié and J. Roberts, "Bandwidth sharing and admission control for elastic traffic," in *Proc. INFOCOM'99*, Mar. 1999, pp. 1395–1403.

# On Learning and the Quality of Service in a Wireless Network

Tiina Heikkinen

Dept. of Mathematical Statistics  
University of Lund, Box 118  
22100 Sweden

**Abstract.** The purpose of this paper is to study the quality of service and optimal distributed resource allocation in a wireless network. A wireless network is similar to the Internet in that the system description must allow for distributed resource allocation under uncertainty. The quality of service that results from the operation of a distributed system is analyzed in terms of a signal-to-noise model of a power-controlled wireless network. The method of analysis is game theory with learning. Recently, the allocation of resources in the Internet has been studied using solution concepts based on learnability. The learnability in the context of a distributed mobile network is studied here. By the appropriate redefinition of the strategy spaces of the users, the resource allocation game, defined between the users, is learnable in the sense that eventually the users learn the optimal strategy concerning resource usage.

**Technical subject area:** Wireless Communications, Quality of Service, Wireless Internet

## 1 Introduction

Future wireless multimedia networks provide new challenges to radio resource management. This paper concentrates on the managing of network resource allocation in a decentralized way. E.g. in packet-switched multimedia networks [1] users often send packets independently of each other. Recently [15] has illustrated how game theory can be used to formulate and analyze decentralized resource allocation in congested networks. The purpose of this paper is to discuss how game theory applies in the analysis of the distributed resource allocation of future multimedia networks, characterized by bursty transmissions across an arbitrarily time-varying channel (like the Internet). In particular, the learnability of the optimal strategies in a time-varying distributed environment is being analyzed.

This work studies the optimal quality of service *QoS* that can be achieved in a distributed wireless network and the mechanisms through which the optimal *QoS* can be reached under uncertainty. The mechanism analyzed in this work consists of a probabilistic learning automation. Under certain assumptions concerning the strategy spaces of the users (players in the network), the users learn the optimal strategies in a noisy

---

<sup>0</sup> Part of this work was carried out while the author was with Rutgers University.

asynchronous network model in bounded time. The general rapidness of convergence under asynchronous strategy choices of users remains an open issue.

By applying game theory under uncertainty, this paper seeks to study the dynamic allocation of resources in a wireless network in the case of uncertain time-varying link gain matrix. The model studied describes the uplink (the case of a downlink is analogous) of a wireless network that consists of  $m$  users transmitting to  $n$  receivers. The paper is organized as follows. Section 2 discusses solution concepts for the distributed dynamic resource allocation of a time-varying network. Section 3 presets a formal model of dynamic distributed resource allocation in a communication network. The extension of the model of distributed resource allocation to a mobile distributed network is discussed in section 4. Section 5 concludes on implementation issues.

## 2 Solution Concepts for a Wireless Network Under Uncertainty

### 2.1 Game Theory Under Uncertainty

This paper discusses game theory under incomplete information in a setting where  $n$  users share bandwidth in a time-varying network. Game theory under incomplete information refers to the situation prevalent in multimedia networks where some or all players (users) lack full information about some aspects of the game such as the other players payoff functions, or even their own payoff functions. This is the situation prevalent in future wireless multimedia networks where the system should operate in a distributed way under incomplete information [1].

The question then is, which are the appropriate solution concepts for a stochastic wireless network administered in a distributed fashion. As discussed in [5, 6] in the case of the Internet, traditional game-theoretical solution concepts need not apply in applications where economic agents know little, and new solution concepts such as those based on learning, are being needed.

There seems to be at least two strands to follow in the development of a new solution concept for future wireless multimedia, where Aumann's "common knowledge" can not be assumed to apply. First, following e.g. [5, 6] a distributed system could be based on a learning approach (section 4). Second, various relaxations of the common knowledge assumption can yield new solutions [10].

### 2.2 Distributed Resource Allocation in a Mobile System

Resource allocation (transmit power allocation) in a wireless network has been studied as a deterministic centralized optimization problem and using stochastic approximation techniques [16]. This section discusses distributed solution concepts for resource management in distributed (decentralized) wireless networks under uncertainty.

Define the quality of service ( $QoS$ ) in terms of signal-to-noise-ratio ( $SNR$ ). The framework of analysis is the  $SNR$ -model [16, 17] of a wireless network. Letting the users be the players and the signal-to-noise ratio of a user define the payoff of the user, the  $SNR$ -model defines an externality game [4]. To see this, first consider the definition of an externality game.

**Definition 1.** A non-atomic game is a game with a continuum of users, in which no single player can affect the other players by changing only her own action. A nonatomic externality game (NAEG) is a nonatomic game with  $m$  classes of players and submodular payoff functions [4].

Here maintain the assumption that no user can affect the other players by changing her action. Here the players are the users who ignore the impact of their choice of the use of resource (transmit power) on the externality (interference) of other users.

The key restriction on a representative user  $i$ 's payoff (signal-to-noise ratio)  $\alpha_i$  is that it only depends on her own strategy  $x_i$  (transmit power) and the externality vector  $f(x)$ . Thus, define [16]

$$\alpha(x, f) = \alpha_i(x_i, f(x)) = \frac{g_i x_i}{f(x)} = \frac{g_i x_i}{\sum_{j \neq i} g_j x_j} \quad (1)$$

Here by definition  $\alpha_i$  is a submodular function i.e. function with decreasing differences: for all  $x, x'$  in the strategy space and  $x > x'$ ,

$$\alpha_i(x, f) - \alpha_i(x', f) = \frac{g_i(x - x')}{\sum_{j \neq i} g_j x_j} \quad (2)$$

is nonincreasing (in fact decreasing) in  $f$ .

Thus, a wireless network is another example of a nonatomic externality game. Hence, learnability in a wireless network can be established by learnability conditions for an NAEG [4]. The main result in [4] for simple NAEGs connects their  $O$ -solvability (solvability by the elimination of overwhelmed strategies,) to dominance- $D$ -solvability and hence to best reply stability;

**Definition 2.** For a given player  $i$ , strategy  $\mathbf{x}$  overwhelms strategy  $\mathbf{x}'$  relative to strategy set  $S$  iff  $\alpha_i(\mathbf{x}, f) > \alpha_i(\mathbf{x}', f')$  for all  $f, f'$ . The set of strategies which are not overwhelmed are denoted by  $O_i(S)$ .

Define  $O^k(S) = O(O^{k-1}(S))$  with  $O^1 = O(S)$ . Note ([4]) that  $O^\infty = \lim_{k \rightarrow \infty} O^k$  is well defined due to the monotonicity of  $O$ .

**Definition 3.** A game is  $O$ -solvable if  $\text{card}(O^\infty) = 1$  for almost all  $i$ .

$O$ -solvability thus is a strong condition requiring that even in an asynchronous game where a player does not know the opponent's strategies, the player still can eliminate dominated strategies.

In order to apply solvability concepts for simple NAEGs it is necessary to redefine the signal to noise model so that it becomes a simple NAEG, i.e. a participation game where players only have two choices, whether or not to participate (now). This redefinition is discussed in section 2.3 below and formalized in section 3.

To motivate the application of solvability concepts for simple distributed NEAGs in the context of a mobile distributed system assume that mobility as such does not change the nature of the underlying distributed externality game. Section 4 discusses the reasonability of this assumption.



### 2.3 Learning Games

Following [5], the following optimization requirements can be placed on a probabilistic learning algorithm:

- When placed in an unknown but eventually stationary environment, the algorithm converges in bounded time, to strategies that maximize average payoff.
- Law of Effect: increasing the payoffs for a given strategy increases the probability of that the strategy being played in the future.

By assumption, for both the standard learning automation and responsive learning automation (a generalization of the former) there is a time  $T$  after which there is a particular strategy whose payoffs are maximal for all histories of the play. For the power control game [10], this assumption does not directly apply, since the deterministic optimum strategy is a mixed strategy [10].

However, by redefining the strategy sets appropriately, the probabilistic learning algorithm becomes directly applicable in distributed resource allocation in a multimedia network, as demonstrated in section 3.2. Section 3.1 first constructs a delay-limited centralized solution to the congestion problem of sharing a limited bandwidth; by adding a temporal dimension, an atemporal solution concept for the congested wireless network can be made comparable to the temporal (delay-based) solution concepts to the congestion problem [15]. Section 3.2 extends 3.1 to a distributed setting.

### 2.4 Mutual Knowledge Games

A simple way to include imperfect information in the definition of an information capacity game between users is to define the information capacity as the outcome of a minimax game, following [10]. The minimax capacity allows to interpret the strategies as mixed strategies. The connection between mixed strategies and incomplete information games is discussed in [12] and the application of the connection to wireless networks in [10].

## 3 A Delay-limited Solution to Resource Allocation

### 3.1 A Centralized Solution to Temporal Resource Allocation

In a model with temporal dynamics, the objective is, in a discretized time model, to maximize, with respect to transmit powers, the service quality as measured by the signal to noise ratio ( $SNR$ )  $\alpha$  at each time interval, taking into account a delay cost formalized below following [8]. A delay-limited capacity is related to the information-theoretic capacity and has recently also been discussed in [2]. A delay-limited capacity has an economic flavor; in [8] and [2] a certain threshold level capacity must be achieved; otherwise the usage of transmit power would be inefficient. Intuitively, when fading is very bad it is better to accept outage [2]; this idea is formalized below following [8].

Here, like in [14], the different rates are modeled by varying the elements in the gain matrix  $\mathbf{G}$  ( $m \times n$  or  $m \times nm$ , where  $m$  is the number of information sources and

$n$  is the number of receivers). In the generalized multiple access model (see appendix C), the coefficient  $g_{i,j} = (\mathbf{G})_{i,j}$  denotes the fading of transmitted information signals by user  $i$  to receiver  $j$  that accommodates transmissions of datatype  $j$ .

Denote the solution sequence by  $\{\alpha_t^*\}_{t=0}^T$ . Formally, this sequence is the solution for (when taking into account the delay cost, formalized in (4) below

$$\max E \sum_{t=0}^T \alpha_t \quad s.t. \quad \mathbf{F}_t \mathbf{x}_t \alpha_t \leq \mathbf{G}_t \mathbf{x}_t \quad (3)$$

Each period the optimal capacity  $\alpha$  is the solution to Problem 1: maximize  $\alpha$  with respect to  $\mathbf{x}$  subject to  $\mathbf{x} \geq 0, \mathbf{x} \leq \mathbf{x}_{max}$  and subject to the constraint, stating the *QoS* requirement,  $\mathbf{G}\mathbf{x} - \alpha\mathbf{F}\mathbf{x} \geq 0$ . The optimal ray solution to the one period problem is given by  $x_i = \frac{1}{g_i}$  and  $p_i = \frac{1}{n}\forall i$ . To model dynamic resource allocation, introduce a parameterized discount delay cost  $0 \leq b \leq 1$  and an additive delay cost  $c \geq 0$ . Let there be two state variables,  $\alpha$  and  $i$ :  $i = 0$  if  $\alpha$  is rejected in the beginning of the period  $t$  (i.e.  $\alpha < \bar{\alpha}$ ),  $i = 1$  if  $\alpha$  is accepted:  $\alpha \geq \bar{\alpha}$ . The Bellman equations for  $v(\alpha, i), i = 0, 1$  are, assuming there is in addition to the delay cost defined by  $c$  below a proportional (to the value function) delay cost component  $b \in (0, 1)$  when deferring the choice of the power vector by  $\Delta(t)$  to period  $t + 1$  with different realizations:  $E(\mathbf{F}_{t+1})$  and  $E(\mathbf{G}_{t+1})$ :

$$\begin{aligned} v(\alpha, 0) &= \max\{\alpha + bEv(\alpha, 1), bEv(y, 0) - c\} \\ v(\alpha, 1) &= \max\{\alpha + bEv(y, 0)\}. \end{aligned} \quad (4)$$

Define  $\bar{\alpha}$  by

$$\bar{\alpha} + bEv(\bar{\alpha}, 1) = bEv(y, 0) - c, \quad (5)$$

where  $\bar{\alpha}$  is the reservation (i.e. threshold) capacity. Choose the foregoing model to define the expected value of capacity as [8]:

$$Ev(\alpha, 0) = \frac{\bar{\alpha}(1+b)}{b(1-b)} + \frac{c}{b(1-b)}. \quad (6)$$

$Ev(\alpha, 0)$  captures the optimal dynamic behavior of the network, since its arguments satisfy the Bellman equation.

### 3.2 Learnability in a Decentralized System

The above section assumed that the distribution  $f$  of  $\alpha = \frac{1}{m-1}$  is stationary. However, this need not be the case for a bursty time-varying multimedia network. A solution concept based on responsive learning automata is discussed next.

Following [5] consider a discrete-time environment with  $M$  possible strategies. Let  $r_i^t$  denote the payoff (capacity) for playing strategy  $i$  at time  $t$ , where as above,  $0 \leq r_i^t \leq 1$ . The state of an automation is a probability vector such that at time  $t$  the automation picks strategy  $i$  with probability  $p_i^t$  at random. To begin with, assume a stationary environment like in section 4 and consider a standard learning automation *LA* (generalized by a responsive one.) A standard parameterized (by  $\beta$ ) updating rule [5] is

$$p_i^{t+1} = p_i^t + \beta r_i^t (1 - p_i^t) \quad (7)$$

$$\forall j \neq i \quad p_j^{t+1} = p_j^t (1 - \beta r_i^t) \quad (8)$$

Assuming the  $r_i^t$  are chosen from some stationary probability distribution so that  $P(r_i^t \leq x)$  is given by  $F_i(x)$  independent of  $t$ . The  $LAs$  are  $\epsilon$ -optimal in that for any  $\epsilon > 0$  there exists some  $\beta$  such that

$$\lim_{t \rightarrow \infty} E\left(\sum_{i=1}^m p_i^t r_i^t\right) > \max(E(r_i^s)) - \epsilon \quad (9)$$

Here consider  $M = 2$  possible strategies: accept  $A$  or reject  $R$  (and carry a delay cost parameterized by  $(c)$  as above). Then  $r_i^t = r^t = \frac{1}{m^t - 1}$ , where  $V$  here is a constant, and the standard updating rule is given by, if strategy  $A$  is picked at time  $t$

$$p_A^{t+1} = p_A^t + \beta \frac{1}{m^t - 1} (1 - p_A^t) \quad (10)$$

$$p_R^{t+1} = p_R^t (1 - \beta \frac{1}{m^t - 1}) \quad (11)$$

Notice that here the payoffs are defined, based on the analysis in 3.1. Then the learning automation defined by (10)-(11) is  $\epsilon$ -optimal [5]: the automation can achieve an asymptotic expected payoff arbitrarily close to the optimal payoff. The play converges to a single strategy.

In an arbitrarily time-varying environment, the collapse to a single strategy is not necessarily reasonable. A responsive learning automation  $RLA$  would require that all strategies are played with strictly positive probabilities:

$$p_A^{t+1} = p_A^t + \beta \frac{1}{m^t - 1} a_R^t p_R^t \quad (12)$$

$$p_R^{t+1} = p_R^t - \beta \frac{1}{m^t - 1} a_R^t p_R^t \quad (13)$$

where  $a_i^t = \min\{1, \frac{p_i^t \beta / 2}{\beta p^t}\}$ . A more general convergence criterion for  $RLAs$  than  $\epsilon$ -convergence is given in [5]. Loosely, convergence for  $RLAs$  is defined so that the probability of playing nonoptimal strategies converges to zero.

## 4 Learnability on a Mobile Distributed Network

Assume the environment becomes stationary after some finite time. Then, as argued in [4], a simple  $NAEG$  such as defined above (by  $M = 2$ ) is  $O$ -solvable which implies that reasonable learners will converge to the unique Nash-equilibrium even in noisy distributed asynchronous settings.

In particular, showing that the participation game defined above is best-reply *BR* stable is sufficient for a simple *NAEG* to be *O*-solvable and thus learnable. This can be done by checking *D*-solvability of the participation game: an externality game is *BR* stable if and only if it is *D*-solvable, i.e. dominance-solvable: solvable by the iterated elimination of dominated strategies [4].

**Definition 4.** For a player  $i$ , pure strategy  $x$  dominates pure strategy  $x'$  iff  $\alpha_i(x, f) > \alpha_i(x', f) \forall f$ .

**Definition 5.** Let  $D^\infty$  denote the set of strategies after infinite iterative elimination of dominated strategies. A game is *D*-solvable if  $\text{card}(D^\infty) = 1$  for almost all  $i$ .

If the utility functions of the users are similar (users value the trade-off between delay and capacity similarly), the users are symmetric with respect to one another in the participation game, and it suffices to study a representative user. An analogous result to Theorem 1 in [4] follows:

**Proposition 1.** A wireless network as defined as a participation game between users is *D*-solvable with a unique Nash-equilibrium.

*Proof.* A strategy profile  $s$  is a Nash equilibrium in pure strategies if for almost all users  $i$  and all  $z$   $\alpha_i(s(i), f(s)) \geq \alpha_i(s(z), f(s))$ . Given the number of users in the system  $m$ , to accept is the unique best reply for a representative user if  $\frac{1}{m-1} \geq \bar{\alpha}$ , where  $\bar{\alpha}$  only depends on given parameters (see (5)).

The direct applicability of learnability for distributed systems to *mobile distributed systems* can be based on viewing the mobile system as intrinsically distributed system; a rationale for this as given in [9] is that formally, the distributed resource allocation problem for a wireless system is very similar to the distributed resource allocation problem for the Internet [11].

The main difference between the resource allocation model for the Internet and the model for the mobile system is that in the latter mobility is taken into account by the variable link coefficients  $0 \leq g_{ij} \leq 1$  between user  $i$  and base  $j$  whereas in the former these coefficients are binary variables. Assuming for simplicity the binary case for the mobile distributed system would be sufficient to guarantee that users with similar utility functions agree with the optimal capacity equalization  $\alpha_i = \alpha$ ; letting each user  $i$  have the choices to transmit or not, i.e.  $x_i = 0$  or  $x_i = 1$ , is then equivalent to the choices being accept or reject the resulting capacity  $\frac{1}{m-1}$ , as above.

Also, in general, when the  $g_{ij}$ 's are not binary variables, the distributed resource allocation game implies capacity equalization if the user optimize utility functions  $u(X)$  with respect to the received signal  $X = gx$  instead of with respect to the energy usage  $x$ . Then again there are two choices at each time period:  $X_i = X^* = u'^{-1}(0) \forall i$  or  $X_i = 0 \forall i$ . In conclusion, learnability for a *mobile distributed network* can be in principle similarly defined as for a distributed network.

## 5 Conclusion: On Implementation Issues

This paper has introduced a framework for the analysis of distributed resource allocation under uncertainty. The paper studied distributed power control in a dynamic wireless network. A solution based on responsive learning automata was suggested.

A probabilistic learning automation is based on a randomized algorithm that can be unlucky, but eventually converges. The two simulation studies commented in [6] are encouraging but not conclusive and can be summarized as follows. The first work on cost sharing games is due to Chen [3] whose results show rapid and robust convergence to the unique Nash-equilibrium in the  $O$ -solvable case. The  $D$ -solvable game, however, was strongly affected by asynchrony and showed less robust convergence. Second, simulations on network games reported by Greenberg, Friedman and Shenker [7] also observed convergence. The discussion in section 4 suggests that these results can be applicable to a mobile distributed network but more simulations are needed to understand the relevance of solution concepts based on learning for a distributed wireless system.

## Appendix A

Let  $M^m \equiv \{\mathbf{x} \in \mathbb{R}_+^m \mid \sum_i x_i = 1\}$ . Second, define the  $n - 1$  simplex:  $M^n \equiv \{\mathbf{p} \in \mathbb{R}_+^n \mid \sum_i p_i = 1\}$ .

The number  $\alpha$  is defined by

$$\alpha = \max_{\mathbf{x} \in M^m} \min_{\mathbf{p} \in M^n} \frac{\sum_{j=1}^n [\sum_{i=1}^m g_{ij} x_i] p_j}{\sum_{j=1}^n [\sum_{i=1}^m f_{ij} x_i] p_j} \quad (14)$$

## Appendix B

Solving for the optimal threshold  $\bar{\alpha}$  gives the following characterization ([8]):

$$\bar{\alpha} = \frac{E(\alpha) - \int_0^{\bar{\alpha}} \alpha f(\alpha) d\alpha - \frac{c}{b}}{\frac{1}{b} + (1 - F(\bar{\alpha}))}. \quad (15)$$

where  $F$  is the cumulative density function of  $\alpha$ . In the absence of the discount cost  $b$  a one period approximation of  $\bar{\alpha}$  is given by

$$\bar{\alpha} = \frac{E(\alpha) - \int_0^{\bar{\alpha}} \alpha f(\alpha) d\alpha - \frac{c}{\tau}}{(1 - F(\bar{\alpha}))}, \quad (16)$$

where  $\tau = T - t$  and capacity  $\alpha_t$  is accepted at time  $t$ .

The expression (16) gives a one-time-stopping rule; applying this repeatedly over time approximates the optimal dynamic resource allocation.

## Appendix C

The key features of a wireless network can be captured using a fairly elementary mathematical model. Typically, both the receivers and transmitters are scattered in some geographical area and the transmitted signals propagate through a physical channel modeling the attenuation of transmitted signal powers through each propagation path. The multiple access protocol defines, together with channel models, the associated interference model. Often interference is modeled as Gaussian noise, which is justified if there are numerous interferers with pseudo-random waveforms and if they are sufficiently weak when compared to the desired user [13].

### *Uplink*

Here we summarize the uplink power control problem and characterize the optimal solution. Consider first an isolated cell with one antenna. We assume that the received signal  $\mathbf{z}$  can be described by a linear model

$$z_i(t) = \sum_{j=1}^m (x_j g_{ij})^{\frac{1}{2}} b_j s_j + n, \quad (17)$$

where  $n$  is the additive Gaussian noise,  $b_j$  denotes information bit transmitted by user  $j$  and  $s_j$  is the signature waveform of user  $j$  [16]. The power control problem is to  $\min \sum x_i$  such that

$$\mathbf{G}\mathbf{x} = \alpha(\mathbf{F}\mathbf{x} + \mathbf{n}), \quad (18)$$

where  $\mathbf{F}$  denotes the interference (externality) matrix. In the absence of external noise, the optimal solution can be characterized as in appendix A [8].

## References

1. F. Borgonovo and M. Zorzi, "Slotted Aloha and CDMA: A Comparison of Channel Access Protocols in Cellular Systems, Wireless Networks", 3, 1997
2. G. Caire, G. Taricco and E. Biglieri, "Optimum Power Control Over Fading Channels", IEEE Transactions on Information Theory, 41, 5, 1999
3. Y. Chen, "Asynchronicity and Learning in Cost Sharing Mechanisms", Mimeo 1996
4. E. Friedman, "Learnability in a Class of Non-Atomic Games arising on the Internet", Mimeo 1998
5. E. Friedman and S. Shenker, "Synchronous and Asynchronous Learning by Responsive Learning Automata", Mimeo 1996
6. E. Friedman and S. Shenker, "Learning and Implementation on the Internet", Mimeo 1998
7. A. Greenwald, E. Friedman and S. Shenker, "Learning in Network Contexts: Experimental Results from Simulations", Mimeo 1998
8. T. Heikkinen, "Capacity and Resource Allocation in a Multi-Media network", IEEE International Symposium on Information Theory, Boston MA 1998
9. T. Heikkinen, "Optimal Quality of Service and Pricing in the Wireless Internet", proc. ITC Specialist Seminar on Mobile Systems, Lillehammer, March 2000
10. T. Heikkinen, "A Minimax Game of Power Control Under Incomplete Information", DI-MACS Technical Report 99-43

11. F. Kelly and R. Gibbens, "Resource Pricing and the Evolution of Congestion Control", <http://www.statslab.cam.ac.uk/frank/evol.html>
12. M. Osborne and A. Rubinstein, *A Course in Game Theory*, MIT Press 1994
13. T. Rappaport, *Wireless communications*, IEEE Press, 1996
14. A. Sampath, P. Kumar and J. Holtzman, "Power Control and Resource Management for a Multimedia CDMA Wireless System", proceedings of PIMRC 1995, Toronto, Canada
15. S. Shenker, "Making Greed Work in Networks", IEEE Transactions on Networking, 3(6), 1995
16. S. Ulukus and R. Yates, "Stochastic Power Control for Cellular Radio Systems", *IEEE Transactions on Communications* 46(6), 1998
17. Zander, J., "Performance of Optimum Transmitter Power Control in Cellular Radio Systems", IEEE Transactions on Vehicular Technology, vol. 41, no. 1, Feb. 1992

# Distributed Fair Bandwidth Allocation of a Wireless Base Station

György Miklós<sup>1</sup> and Sándor Molnár<sup>2</sup>

<sup>1</sup> Traffic Analysis and Network Performance Lab, Ericsson Research  
1037 Budapest, Laborc u. 1., Hungary  
`Gyorgy.Miklos@eth.ericsson.se`

Tel: +36 1 437 7633, Fax: +36 1 437 7219

<sup>2</sup> High Speed Networks Lab, Dept. of Telecommunications and Telematics,  
Technical University of Budapest  
1117 Budapest, Pázmány P. sétány 1/D  
`molnar@ttt-atm.ttt.bme.hu`

**Abstract.** We present a novel approach to wireless fair scheduling in a distributed architecture. Our scheme is based on a simple extension of wireline fair scheduling by compensation for lost capacity after the losses occur. In addition, the scheduler can give limited compensation for unused capacity. This provides an incentive for users of the location-dependent wireless channel to optimize their channel usage on their own. In this way the scheduler does not need to be aware of the state of the wireless channel for each user. We present a mechanism for user behaviour and illustrate our scheme by simulation results.

## 1 Introduction

Motivated by the growing use of wireless access technologies, the adaptation of wireline fair queuing algorithms to wireless environments has received increasing attention. Often it is the wireless link that becomes the bottleneck of an end-to-end flow which further emphasizes importance of the topic.

Wireless fair scheduling is different from wireline fair scheduling in several aspects. The scheduler has to take into account not only the quality of service requirements of the individual flow, but also the fact that the quality of the wireless channel may change over time and errors are typically location-dependent. Furthermore, the MAC layer of the specific link may impose additional constraints.

A number of recent papers deal with the adaptation of wireline fair scheduling algorithms to a wireless environment (e.g. [5], see [4] for a comparative overview). They are based on the assumption that errors are typically bursty in nature, such that it is possible for the scheduler to predict the state of the channel for each user. As long as the number of errors are bounded from above, a minimum throughput and a maximum delay guarantee can be given to a flow.

In this paper we start from different assumptions and arrive at a different architecture. We do not assume any bound on the number of errors and accordingly suppose that the quality of the wireless channel will be visible to the end



applications that can adapt to the changes in the service quality. Our purpose is not to hide the effects of wireless channel errors but to give partial compensation so that the otherwise unfair service (due to location-dependent errors) is made fairer.

Observe that bandwidth allocation over a lossy channel introduces a fundamental trade-off between fairness and utilisation: users with higher error rates need to be allocated more capacity to get the same fair amount of service but this decreases overall utilization; and vice versa: utilization is increased by allocating more capacity to users with better channels, which makes the allocation less fair. The actual amount of compensation is based on the service provider's choice in this trade-off. Our aim here is to extend wireline fair scheduling algorithms by a simple mechanism for compensation to improve fairness, so that this tradeoff between fairness and utilisation can be controlled by the service provider.

In [3] we introduced a simple compensation mechanism where the master scheduler in the base station compensates for losses over the air interface *after they occur*. We analysed the trade-off between fairness and utilisation and the impact of link layer ARQ and TCP traffic.

Our approach here is an extension of our previous work. The master scheduler in the base station does not have any information or assumptions about the state of the wireless channel. Instead it compensates for channel errors after they occur. We extend the architecture by *allowing the users to defer their transmission to a later time when the channel is temporarily in a bad state*. Our approach is therefore decentralized in the sense that the master scheduler using a simple compensation mechanism is making the scheduling decision without any regard to the wireless channel state. Each user of the channel is responsible by itself for the estimation and prediction of its own channel, and can optimize when to transmit on its own. When the channel is expected to be bad for some time, the user can defer its transmission until the channel is expected to recover, based on the measured channel properties. This is encouraged (but not controlled) by the master scheduler as it allows the users to partially reclaim unused capacity in the future.

This approach offers several advantages. First, we do not need to make any assumptions about the error characteristics of the channel and the master scheduler does not need the prediction of the state of the link. This is useful since it is generally difficult to reliably predict the channel state in the future. Second, our scheme offers a modular implementation and greatly simplifies the master scheduler. Third, it offers a decoupling of functionality: the users' estimation and prediction of the channel can be changed without modifying the master scheduler.

The paper is organized as follows. Section 2 presents the system architecture. The master scheduling algorithm is described in Section 3. Section 4 shows how a user can optimize for itself the channel usage. Our simulation results are presented in Section 5, and Section 6 concludes the paper.

## 2 System Architecture

In this subsection we briefly present the system architecture in which we applied our scheme. An example to such a system is the HiperLAN type 2 wireless LAN system [2], but the compensation algorithms can be adapted to other architectures as well.

Wireless access is provided in a cellular architecture, where an Access Point (AP) serves as the base station (BS) in each cell. APs are connected by a fixed network. Mobile Terminals (MTs) are associated with the AP of the cell they are in. All communication from and to a MT takes place through the AP.

The system uses TDMA/TDD (Time Division Multiple Access/Time Division Duplex) access, that is, all communications in a cell, both uplink and downlink, use the same frequencies, and are multiplexed in time as determined by the AP. The MAC protocol is frame-based, with a fixed-sized frame of 2ms. User data is transmitted in fixed-sized radio packets (referred to as PDUs, protocol data units) with 48 bytes of payload. Lost packets may be retransmitted as determined by the ARQ (automatic repeat request) protocol.

Each frame consists of a downlink and an uplink part. At the beginning of each frame, the AP announces how uplink and downlink transmissions are allocated within the current frame. Therefore communication is contention-free, with the exception of a short random access channel at the end of each frame, where the MTs can transmit control information to the AP.

The process of scheduling transmission resources in a MAC frame takes place as follows. The master scheduler within the AP takes as its input the number of packets queued for transmission for each user of the air interface. For downlink traffic this is readily available at the AP, while for uplink traffic this information is sent to the AP in control messages.

The master scheduler can run its scheduling algorithm frame by frame to determine how the resources are allocated. It takes as input the resource requests from users transmitter either in the random access channel at the end of frames or earlier in the frame as control information. Depending on the implementation the scheduling process can incur one or more frames of delay which is not considered here. The output of the scheduler are the resource grants announced at the beginning of the new frame.

When applying a fair queuing algorithm in the master scheduler, we can think of the scheduling process in two steps. Based on the resource requests from the users, a virtual server running a fair queuing algorithm serves the users in the first step, and the amount of total service is determined for each user for the next frame. This is the amount of allocation each user will get in the frame, but the actual ordering of allocation within the frame may be changed in the second step. Most likely allocations for a given user will be grouped together to reduce overhead, and downlink and uplink transmissions will also be grouped together. We address only this step in this paper, so it must be kept in mind that the actual transmissions may take place at a time of maximum one frame apart from when the server made the service.

### 3 Master Scheduling Algorithm

There are a number of wireline fair scheduling algorithms such as Weighted Fair Queuing (WFQ) and others [6]. We have chosen Start-time Fair Queuing (SFQ) for our scheme, which will be motivated in Subsection 3.2. First, we introduce SFQ.

#### 3.1 Start-Time Fair Queuing

Start-time Fair Queuing [1] greatly reduces the computational complexity of WFQ by avoiding the need to simulate the fluid server in real time. Virtual time in SFQ is derived from the start tag of the packet in service. Another advantage of this method is that SFQ is applicable to variable rate servers without a need to take the server rate into account in the virtual time computation. The price paid for this simplicity is that the delay guarantee increases with number of flows. The fairness, throughput and delay properties of SFQ are analyzed in [1].

Start-time Fair Queuing is defined as follows [1]:

- 1 Packet  $j$  of flow  $i$  is stamped with a start and finish time  $S_i^j$  and  $F_i^j$ , respectively, upon arrival at time  $A_i^j$ .

$$S_i^j = \max\{v(A_i^j), F_i^{j-1}\}, \quad F_i^j = S_i^j + \frac{L_i^j}{w_i} \quad \text{for } j \geq 1 \quad (1)$$

where  $F_i^0=0$ ,  $w_i$  is the weight of flow  $i$ ,  $L_i^j$  is the length of packet  $j$  of flow  $i$ , and  $v(t)$  is the virtual time at time  $t$ .

- 2 The server virtual time is initially 0. During a busy period the server virtual time at time  $t$ ,  $v(t)$ , is defined to be equal to the start tag of the packet in service at time  $t$ . At the end of a busy period,  $v(t)$  is set to the maximum of finish tag assigned to any packets that have been serviced by time  $t$ .
- 3 Packets are served in increasing order of start tags; ties are broken arbitrarily.

#### 3.2 Fair Queuing in the AP Master-Scheduler

It is possible to use any of the fair scheduling algorithms in the first step of the AP master scheduler [6]. A number of differences from wireline fair queuing must be noted however: (a) Scheduling has a granularity of a *user*, which may be coarser than the granularity of *flows*. (b) The scheduling process has no immediate information about the arrival of packets or the immediate backlogged status of queues. All that the server can take into account is whether users are satisfied or unsatisfied as compared to the resource requests that users make on a frame by frame basis. This is why we use the terms satisfied or unsatisfied, instead of backlogged or unbacklogged, for the status of the users in the scheduling process. (c) The scheduler is only responsible for the amount of allocations to the users, and the users of the air interface themselves can decide how to utilize the allocated capacity for new transmissions and retransmissions. Since the scheduler

is in no control of the individual packets, it is better to use per user state information in the scheduling process, rather than per packet information (as is traditional with the start and finish time tags in fair queuing algorithms). (d) The computation of the system virtual time (i.e. the simulation of the fluid server) is made easier by the fact that users can become unsatisfied (i.e. equivalent of backlogged) only at frame boundaries, which must be also packet service boundaries. This follows that the satisfied status of a user changes only after the service of a packet. (e) The unsatisfied status of a user may change to satisfied even without the requested amount of service provided by the scheduler because the user may give up further transmission attempts of some packets at the expiration of some timer, or because the user may decide to defer its transmission to a later frame, as discussed later. When the virtual fluid server is simulated, this requires modifications since the set of unsatisfied users can change in the past in the sense that virtual service was given to a user but the corresponding real service cannot be given later. To solve this problem, the real service to a user that is no longer unsatisfied can be given to another user as a "present". This is not discussed further here.

Here we apply SFQ to our study. The choice is motivated by the fact that SFQ is not sensitive to the changes of the satisfied status of the users, i.e. the computation of the virtual time or the selection of the next user does not have to be modified as users become satisfied and unsatisfied (see (e) above), and that SFQ is computationally simple, yet provides fairness, throughput and delay guarantees. Although a computationally more expensive algorithm such as WFQ could provide tighter delay bounds, the frame-based architecture itself introduces a delay in any case which does not motivate a computationally more expensive algorithm.

We adapt SFQ to the master-scheduler of the AP in the following way.

- 1 A start and a finish tag,  $S_i$  and  $F_i$ , are associated with each *user*, corresponding to the virtual start and finish time of the packet at the head of the queue. When a new packet enters the head of the queue at time  $t$  (i.e. a packet has been served, or the user becomes unsatisfied), then the new values are computed from the old value of the finish time,  $F'_i$ , as

$$S_i \leftarrow \max\{F'_i, v(t)\}, \quad l \leftarrow \frac{L}{w_i}, \quad F_i \leftarrow S_i + l \quad (2)$$

where initially  $F'_i = 0$ ,  $w_i$  is the weight of flow  $i$ ,  $L$  is the length of all packets,  $l$  is the normalized length of the packet, and  $v(t)$  is the virtual time at time  $t$ .

- 2 The server virtual time is initially 0. During a busy period the server virtual time at time  $t$ ,  $v(t)$ , is defined such that when a packet from user  $i$  enters service, the system virtual time becomes  $S_i$ , the start tag corresponding to the packet. When the system is idle the virtual time is increased linearly such that  $dv(t)/dt = C/\sum_{i \in \mathcal{U}} w_i$  where  $\mathcal{U}$  is the set of all users.
- 3 Packets are served in increasing order of start tags; ties are broken arbitrarily.

Note that the start and finish tag computation was modified in so far as only the start and finish tag of the first packet of a user is maintained. (But even when the last packet of a user is served, its finish tag, denoted by  $F'_i$ , must be maintained.) The virtual time computation was modified in that for an idle server, the virtual time is incremented linearly so that it reflects the increase in the minimum amount of virtual service that each user could have been given if it were unsatisfied (as will be discussed later).

### 3.3 Compensation of Lost and Unused Resources

Location dependent errors on the wireless channel can alter the amount and distribution of resources utilized by the users, which motivates the need for a compensation mechanism in the scheduler. We extend the scheduler by two types of compensation: *compensation of lost and unused resources*.

Lost resources correspond to the packets which were not received correctly. They can become known to the AP scheduler through the ARQ protocol after the errors occur. By compensating for lost resources the differences of user goodput due to the different radio channel conditions can be reduced.

Unused resources correspond to the amount of service a user could have got while it did not request resources. By compensating unused resources later we give incentive to the users to monitor the state of their wireless links themselves and defer transmission when the link is temporarily in a bad condition.

To implement the compensation, we introduce the state variable *lag* for each user, denoted by  $n_i$ , to represent the amount of normalized service that the user should get in compensation. The constant  $\beta_l$ ,  $0 \leq \beta_l \leq 1$  represents the amount of compensation for lost resources.  $\beta_l = 1$  means that all of the lost capacity is compensated later,  $\beta_l = 0$  means no compensation for lost capacity. Similarly, the constant  $\beta_u$  represents the amount of compensation for unused resources,  $0 \leq \beta_u \leq 1$ . The constant  $\gamma$ ,  $0 \leq \gamma \leq 1$ , is called the speed of compensation, and determines the increase of allocations when a user is being compensated.  $\gamma = 0$  corresponds to no compensation, whereas  $\gamma = 1$  corresponds to immediate compensation, where a user is compensated before any other users can get more allocations.

The scheduling algorithm is extended as follows.

- 4 After each error of a packet of length  $L$  on the wireless channel for user  $i$  as reported by the ARQ protocol, its lag is incremented by  $\beta_l$  times the normalized service that was lost:

$$n_i \leftarrow \min\{n_i + \beta_l L/w_i, n_{max}\}.$$

- 5 When user  $i$  becomes unsatisfied at the beginning of a frame at time  $t$  after being satisfied, its lag is incremented by  $\beta_u$  times the normalized service that the user missed:

$$n_i \leftarrow \min\{n_i + \beta_u \max\{v(t) - F'_i, 0\}, n_{max}\}$$

where  $v(t)$  is the virtual time at time  $t$ , and  $F'_i$  is the finish time of user  $i$  before it became satisfied. Since  $v(t)$  can be interpreted as the normalized fair amount of service that each user could have received up to time  $t$ ,  $v(t) - F'_i$  is the amount of normalized service that the user missed while it was satisfied. 6 Equation eq. (2) is modified as follows.

$$l \leftarrow \frac{L}{w_i}, l_c \leftarrow \min\{n_i, \gamma l\}, F_i \leftarrow S_i + l - l_c, n_i \leftarrow n_i - l_c, \quad (3)$$

where  $l$  is the normalized length of the packet,  $l_c$  is the normalized compensation given during the service of the packet.

This means that while a user is being compensated, the normalized service given when a single packet is served is artificially decreased by  $\gamma$  times the normalized length of the packet. This can also be interpreted as increasing the weight of the user from  $w_i$  to  $w_i/(1 - \gamma)$ . The lag is decreased by the amount that was used up for the compensation.

The amount of lag is maximized by  $n_{max}$ . The purpose of this limit is to allow the compensation of lost and temporarily deferred transmissions but limit the amount of compensation for a users with little or no traffic for a long period of time.

As seen from the implementation of compensation a user is being compensated in such a way that its weight is in effect increased from  $w_i$  to  $w_i/(1 - \gamma)$ . This is why the admission criterion for a new flow (which is in the case of SFQ  $\sum_{i \in \mathcal{U}} w_i \leq C$ ) becomes  $\frac{1}{1-\gamma} \sum_{i \in \mathcal{U}} w_i \leq C$ .

## 4 User Behaviour

The previous section has presented the master scheduling algorithm that provides two kinds of compensation to the users: compensation for lost and unused resources. Each user can observe the quality of its own channel through measurements and can decide how much capacity to request. Here we make the following simplifying assumptions: we assume that the success of the transmissions of a user is known immediately after the transmissions through the use of an ARQ mechanism on the link layer, there is no delay associated with the capacity requests, and that resource requests are never lost. A user either makes a resource request according to the packets waiting for transmission in its buffers, or makes a resource request of zero, thereby relinquishing service to a later frame. Our purpose in this section is to arrive at a method for a user to decide when to relinquish service (i.e. make a resource request of 0). We assume that a user has knowledge of the scheduling algorithm and its constant parameters.

Relinquishing service in a given frame can be useful for the mobile because channel behaviour is typically positively correlated, so when a user observes bad channel, it is likely that the channel will continue to be bad for some time.

In the following subsections, we present a simple solution to this problem where the user builds a model of the channel estimating the parameters, which enable it to make a prediction of the future expected channel state and decide when to relinquish service.

#### 4.1 Channel Model and Prediction

We use a simple AR(1) (autoregressive) model. This is a very simple model, yet powerful enough to capture the correlated nature of the channel. Note that the AR(1) model and the quantitative approximation must be regarded as heuristics since we have no way of getting any a priori information on the channel properties and its stationary nature.

We model the success rate in every frame, that is, the fraction of successfully received PDUs out of the transmitted PDUs in every frame. The distribution of errors within a frame is not modeled.

The AR(1) process (established independently for each user) is written as

$$v[t+1] = \rho v[t] + \sigma z, \quad s[t] = \pi + v[t] \quad (4)$$

where  $v[t]$  is the actual AR(1) process, and  $s[t]$  is the process of success rate.  $t$  is the frame counter (integer),  $z$  is a standard normally distributed random variable,  $\rho, \sigma, \pi$  are the parameters of the process. We make the simplification that we do not consider to be part of the AR(1) process those frames where the user is not scheduled.

The expected value, variance and correlation of the process  $s[t]$  is given as

$$E s[t] = \pi, \quad \text{Var } s[t] = \frac{\sigma^2}{1 - \rho^2}, \quad \frac{\text{Cov}(s[t], s[t+k])}{\text{Var } s[t]} = \rho^k \quad (5)$$

We note that the process  $s[t]$  could take on values outside the interval  $[0, 1]$ . However, we are going to use the model only to predict the expected value of the success rate in the future. According to Equation 10 in Subsection 4.1, it will be clear that the prediction cannot take on a value outside the interval  $[0, 1]$  provided that  $\pi$  and the measured values of the process are within that interval.

We need to estimate the parameters of the model in such a way that as more and more measurements are available, the accuracy improves, on the other hand, the parameter estimation adapts to long-term, non-stationary changes in the channel behaviour. To achieve this, we use exponential moving averages in all the parameter estimations with weight  $\phi$ .

Estimations are based on  $s_i^*[t]$ , success rate measured in the last frame.  $\pi$  is estimated at frame  $t$  as

$$\pi[t] \leftarrow \pi[t-1]\phi + s_i^*[t](1-\phi) \quad (6)$$

Estimation of correlation is made indirectly through the estimation of second moment,  $\delta$ , and first order mixed second moment (i.e.  $E(s[t]s[t-1])$ ),  $\psi$ , of the process:

$$\delta[t] \leftarrow \delta[t-1]\phi + (s^*[t])^2(1-\phi), \quad \psi[t] \leftarrow \psi[t-1]\phi + s^*[t]s^*[t-1](1-\phi) \quad (7)$$

The variance of the measured process is estimated as

$$\mu[t] = \delta[t] - (\pi[t])^2 \quad (8)$$

The following estimator can be shown to converge to the correlation if the measured process is AR(1):

$$\rho[t] = \frac{\psi[t] - \pi[t]\pi[t-1]}{\sqrt{(\delta[t-1] - (\pi[t-1])^2)(\delta[t] - (\pi[t])^2)}} \quad (9)$$

Given a current measurement of the success rate  $s^*[t]$  and the estimations  $\pi[t]$  and  $\rho[t]$ , the success rate for the frame  $t+k$  can be predicted as

$$\hat{s}[t+k] = (s^*[t] - \pi[t])(\rho[t])^k + \pi[t] \quad (10)$$

## 4.2 User Decision

Based on the channel measurements and predictions, a user must decide whether to relinquish transmission in the next frame in the hope of more efficient transmission later. Our criterion aims at maximizing throughput for the user. (Here we do not consider explicit delay guarantees for a user, though they may be incorporated as well by imposing additional rules that do not allow a user to defer transmission when the delay bound could be violated.)

In order to be able to provide a decision function that is applicable by a user without any explicit information about other users or the future, we make several simplifying assumptions. We assume that compensation is not yet limited by the maximum lag; and we do not consider changes in the success rate during compensation for unused service; furthermore we approximate the service given to a user for a compensation of  $x$  to be  $x\pi[t]$ , that is, success rate during compensation is approximated by the estimated average success rate  $\pi[t]$ . Our decision will be based on the expected service with compensation for one PDU in a frame.

If the user decides to transmit in the next frame, the expected service per packet in the frame is  $L\hat{s}[t+1]$  (where  $L$  is the packet length), and the expected lost service is  $L(1-\hat{s}[t+1])$ . The expected compensation is  $L(1-\hat{s}[t+1])\beta_l$  giving an expected service of  $L(1-\hat{s}[t+1])\beta_l\pi[t]$  since we assume that the success rate during compensation for lost resources is  $\pi[t]$ . So the expected service per PDU is  $L(\hat{s}[t+1] + (1-\hat{s}[t+1])\beta_l\pi[t])$ . If the user defers transmission to a later frame  $t+d$ , the total service received instead of the allocation of a PDU in the current frame is less by the factor  $\beta_u$  since the service is being compensated for unused capacity. So the expected service per PDU is  $L\beta_u(\hat{s}[t+d] + (1-\hat{s}[t+d])\beta_l\pi[t])$ . Transmission is relinquished in the current slot if the expected service per PDU is increased:

$$L(\hat{s}[t+1] + (1-\hat{s}[t+1])\beta_l\pi[t]) < L\beta_u(\hat{s}[t+d] + (1-\hat{s}[t+d])\beta_l\pi[t]) \quad (11)$$

which gives

$$\frac{\hat{s}[t+1]}{\beta_u} + \frac{\frac{1}{\beta_u} - 1}{\frac{1}{\beta_l\pi[t]} - 1} < \hat{s}[t+d] \quad (12)$$



Deferring transmission in the current frame is useful when the channel model suggests that at a later frame  $t + d$  the expected success rate fulfills the above equation.

We introduce a threshold-based decision criterion for a user using an upper threshold  $\theta_1$  and a lower threshold  $\theta_2$  on the expected success rate for the next frame  $\hat{s}[t + 1]$ . We assume a greedy user meaning a user that has always data to transmit, so that it can use any additional capacity provided to it. (TCP traffic is an example of greedy traffic.)

The upper threshold determines the sensitivity of the algorithm and it is specified through the constant  $\omega$  which gives the thresholds relative position with respect to the average success rate and its standard deviation:

$$\theta_1 = \pi[t] - \omega\sqrt{\mu[t]} \quad (13)$$

The lower threshold is determined in such a way that if the expected success rate in the next frame falls below the lower threshold and the user waits until the expected success rate reaches the upper threshold, then the user is expected to receive more service compared to not deferring transmission. This follows that the relationship between  $\theta_1$  and  $\theta_2$  can be obtained by applying eq. (12):

$$\theta_2 = \beta_u \left( \theta_1 - \frac{\frac{1}{\beta_u} - 1}{\frac{1}{\beta_l \pi[t]} - 1} \right). \quad (14)$$

The rule is that we begin deferring transmission when the expected success rate falls below  $\theta_2$ , and re-start when the expected success rate is above  $\theta_1$ .

### 4.3 Adaptive Threshold Calculation

The previous subsection described how the thresholds  $\theta_1$  and  $\theta_2$  can be computed given  $\omega$ , which determines sensitivity of the thresholds. A high value of  $\omega$  results in lower thresholds and therefore less frequent relinquishing of service. A low value of  $\omega$  on the other hand causes the user to relinquish service more often.

To reach the highest throughput and most efficient resource utilisation, a user should relinquish service as often as possible so that during transmission, the channel behaviour is as good as possible. However, the speed of compensation in the master-scheduler (determined by the parameter  $\gamma$ ) poses an upper limit on the frequency of relinquishing service: the average rate of necessary compensation generated by the user should not exceed the rate of compensation that the scheduler can provide.

The optimal value of  $\omega$  depends on the speed of compensation,  $\gamma$ , but also on the behaviour of other users, whether they request resources or not and how much compensation they receive. This is why we have chosen to implement an adaptive algorithm for setting  $\omega$  based on feedback from the scheduler in the AP. This adaptive algorithm can be regarded as optional in the scheme: without it, a value for  $\omega$  can be set as a constant.

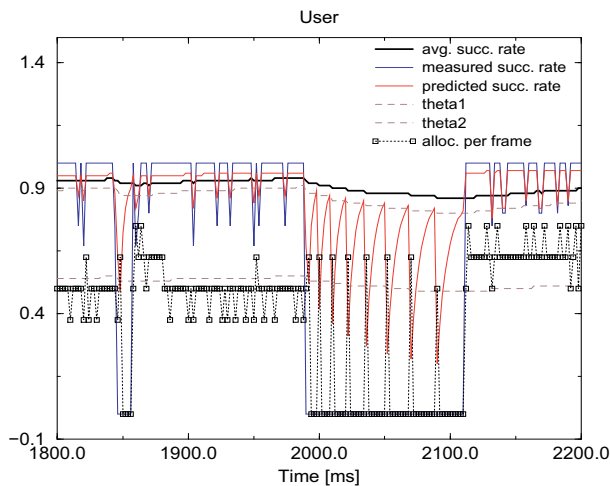
The algorithm works as follows. In the broadcast channel of each frame, the AP gives a one-bit feedback to all the users which is set when the user has

nonzero lag. Each user monitors this bit and computes an exponential moving average (denoted by  $f_i$ ) for the ratio of frames where the user has nonzero lag at the scheduler. When  $f_i$  is too small this means that the scheduler can still provide more compensation. In this case the user decreases  $\omega$  in order to get more compensation for unused resources, so that user is active when the channel is better. When  $f_i$  is too large this means that there is a danger that the lag is too high and reach its upper limit, i.e. the scheduler can not provide the required amount of compensation. In this case  $\omega$  is increased.

## 5 Simulation Study

We have implemented our scheme in a packet-level simulation environment. The simulated architecute conforms to the architecture presented in Section 2 with frame based MAC protol with a frame length of 2ms. In our simulations, the capacity of the system was 8 PDUs/frame where a PDU can carry 48 bytes, giving a total system capacity of 1.5Mbps. This capacity is shared by two users, each having an average PDU loss rate of 25%, but independent channel error models. User 1 has independently distributed losses, while User 2 has bursty losses according to an ON-OFF Markovian model. The average OFF period is 100 PDU transmission times, the loss rate is 95 % in the OFF period, 5% in the ON period. Both users are greedy.

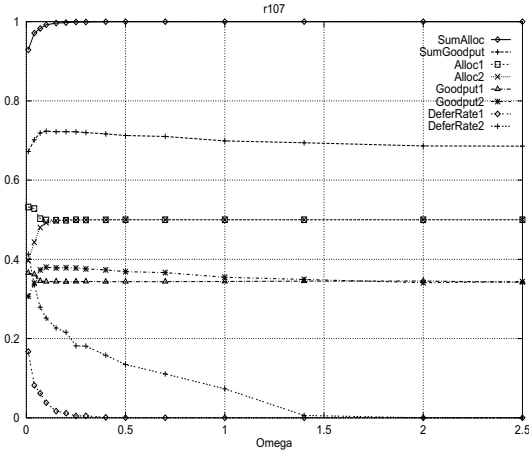
Figure 1 shows a trace from a simulation for User 2, having bursty losses. The figure plots the time evolution of a number of parameters computed each frame. The measured success rate is the ratio of the successful PDU deliveries to all PDU transmissions in the last frame. The average and predicted success rates are computed as described in Subsection 4.1. The thresholds  $\theta_1$  and  $\theta_2$  (Subsection 4.2) are also plotted. The user's



**Fig. 1.** User behaviour when the channel has bursty errors.

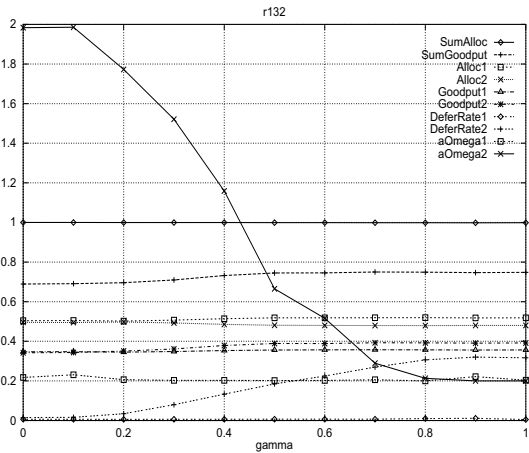
decision and the behaviour of the scheduler is seen on the amount of allocation per frame. In the figure, this gives the number of PDUs allocated in the frame to the user as a fraction of the total number of PDUs per frame (8 in our setup). When the user decides to relinquish service in a frame, this is seen by the lack of allocation per frame.

The simulation illustrates the user behaviour and compensation (we used the parameters  $\gamma = 0.5, \beta_l = 0.5, \beta_u = 0.8$  here). In the figure we can see two periods when the user relinquishes service, one at 1850ms, the other from 1980ms to 2120ms. The latter is a long OFF period in which the user re-starts several times. Note the increased amount of allocation after a user relinquished its service as a result of compensation for unused capacity in the scheduler.



**Fig. 2.** Effect of  $\omega$ ,  $\gamma = 0.5$ ,  $\beta_u = 1$ ,  $\beta_l = 0$ .

(Alloc1 and Alloc2) and the goodput (successfully carried traffic) for the two connections. The graphs DeferRate1 and DeferRate2 show the fraction of frames where User 1 and 2 decided to relinquish transmission.



**Fig. 3.** Effect of  $\gamma$ ,  $\beta_u = 1$ ,  $\beta_l = 0.5$ .

relinquishes more service that the scheduler can compensate. As a result, the allocation to User 2 is decreased, and this, of course, results in goodput decrease.

Figure 2 shows the effect of  $\omega$  on the throughput of connections and system utilisation. In this simulation there was full compensation for unused capacity ( $\beta_u = 1$ ), no compensation for lost capacity ( $\beta_l = 0$ ) and the speed of compensation  $\gamma = 0.5$ . The figure shows the following throughput quantities normalized by the system capacity: the total allocation for both connections (SumAlloc), the total goodput of the two connections (SumGoodput), the allocated capacity from the two connections

The figure clearly shows a maximum for the total goodput and the goodput for User 2 (the one with bursty channel loss model) at near  $\omega = 0.15$ . User 1 can not achieve any improvement since its channel losses are independent. For higher  $\omega$ , the gain of User 2 is less because the user relinquishes transmission less frequently, therefore it does not avoid all the bad channel states that it possibly could. This clearly shows the advantage of using the compensation method for unused capacity. For smaller  $\omega$ , the user

Figure 2 illustrates that there exists an optimal value of  $\omega$  motivating the use of the adaptive  $\omega$  calculation approach described in Subsection 4.3.

Using that algorithm to set  $\omega$  adaptively, Figure 3 shows the effect of changing the speed of compensation,  $\gamma$ , while  $\beta_u = 1$  and  $\beta_l = 0.5$ . (The average value of  $\omega$  for the users are shown by  $\text{aOmega1}$  and  $\text{aOmega2}$ .) As the speed of compensation increases, there is time for more frequent relinquishing of service for User 2 (increase of  $\text{deferRate2}$ ). This is achieved through the decrease of  $\omega$ . Note that as  $\gamma$  increases from 0 to about 0.6, there is significant increase in the total system goodput as well as the goodput of User 2 and also User 1. The increase of goodput for User 2 is explained by its better utilisation of the allocated capacity, while the slight increase of goodput for User 1 is explained by the increase of allocation provided to it due to the compensation for lost capacity.

## 6 Conclusions

We have presented a scheme for the resource allocation of the capacity of a frame-based wireless base station with its performance evaluation. The new contributions of our work are (a) presenting a modular architecture where the fair master-scheduler does not explicitly take into consideration the channel state of individual users; (b) showing how a wireline fair scheduling algorithm can be simply extended to compensate for lost capacity after losses occur; (c) extending the master-scheduler with compensation for unused capacity, so user can optimize for its own when to relinquish service in the hope of more service later; (d) showing one possible way of user behaviour; and (e) showing by simulation that the pair of master scheduler and user decision rule can work together to improve both the total system utilization and the goodput of individual users.

## References

1. Pawan Goyal, Harrick M. Vin, and Haichen Cheng. Start-time fair queuing: A scheduling algorithm for integrated services packet switching networks. *IEEE/ACM Transactions on Networking*, 5(5):690–704, October 1997.
2. HiperLAN/2 Global Forum. <http://www.hiperlan2.com/>.
3. György Miklós and Sándor Molnár. Fair allocation of elastic traffic for a wireless base station. In *IEEE GLOBECOM'99*, December 1999.
4. Thyagarajan Nanadagopal, Songwu Lu, and Vaduvur Bharghavan. A unified architecture for the design and evaluation of wireless fair queueing algorithms. In *ACM MOBICOM'99*, August 1999.
5. T. S. Eugene Ng, Ion Stoica, and Hui Zhang. Packet fair queueing algorithms for wireless networks with location-dependent errors. In *INFOCOM'98*.
6. Hui Zhang. Service disciplines for guaranteed performance service in packet-switching networks. *Proceedings of the IEEE*, 83(10), October 1995.

# Fast Approximate Algorithms for Maximum Lifetime Routing in Wireless Ad-hoc Networks \*

Jae-Hwan Chang and Leandros Tassiulas

Department of Electrical & Computer Engineering  
and Institute for Systems Research  
University of Maryland, College Park MD 20742, USA,  
{jhchang,leandros}@isr.umd.edu

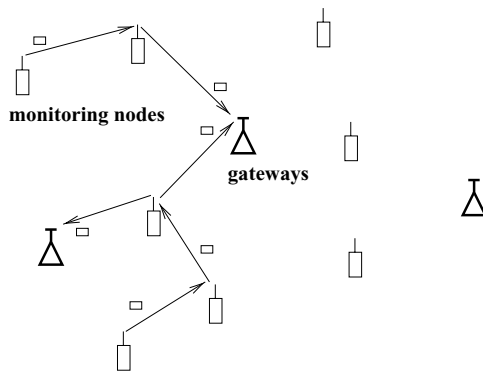
**Abstract.** In wireless ad-hoc networks, routing and power consumption are intrinsically connected since the transmit power level is adjusted depending on the location of the next hop node. Our objective is to route the traffic such that the system lifetime is maximized. In the previous works, we have formulated the problem and proposed heuristic algorithms. In this paper, fast approximate algorithms are proposed both in single commodity case and in the multicommodity case along with the analysis which enables us to control the trade-off between the performance bound and the running time. The problem is then extended to have an additional constraint on the delay.

## 1 Introduction

Consider a group of randomly distributed wireless static nodes as in Fig. 1, where each node has a limited battery energy used mainly for the transmission of data. Assume that at each node some type of information is generated, which needs to be delivered to some nodes designated as gateway nodes. An example may be a wireless sensor network where the sensors gather acoustic, magnetic, or seismic information and send the information to its gateway node which has more processing power for further processing of the information or has larger transmission range for the delivery of the information to a possibly larger network for retrieval by a remote user. The wireless nodes are assumed to have the capability of packet forwarding, i.e., relaying an incoming packet to one of its neighboring nodes, and the transmit power level can be adjusted to a level appropriate for successful reception if the receiver is within the transmission range. Upon or before a new arrival of information either generated at the node itself or forwarded from the other nodes, routing decision has to be made so that the node knows which of its neighboring nodes to send its data to. Note that the routing decision and the transmission energy level selection are intrinsically

---

\* Prepared through collaborative participation in the Advanced Telecommunications/Information Distribution Research Program (ATIRP) Consortium sponsored by the U.S. Army Research Laboratory under the Federated Laboratory Program, Cooperative Agreement DAAL01-96-2-0002.



**Fig. 1.** A multi-hop wireless ad-hoc network is depicted where the information generated at the monitoring nodes are delivered to the gateway nodes.

connected in this power-controlled ad-hoc network since the power level will be adjusted depending on the location of the next hop node.

Most of the previous works on routing in wireless ad-hoc networks deal with the problem of finding and maintaining correct routes to the destination during mobility and changing topology [2,6,12]. In [2,6], the authors presented a simply implementable algorithm which guarantees strong connectivity and assumes limited node range. Shortest-path algorithm is used in this strongly connected backbone network. In [12], the authors developed a dynamic routing algorithm for establishing and maintaining connection-oriented sessions which uses the idea of predictive re-routing to cope with the unpredictable topology changes. Some other routing algorithms in mobile wireless networks can be found in [10,13,14,15], which, as the majority of routing protocols in mobile ad-hoc networks do, use shortest-path routing where the number of hops is the path length.

The problem of minimum energy routing has been addressed before [2,6,7], [8,11,16,17,18]. The approach in these works was to minimize the consumed energy to reach the destination. In [3], knowing that the approach doesn't provide an optimization in the network level, we have formulated the maximum system lifetime routing problem as a linear programming problem, and proposed heuristic algorithms which showed close-to-optimal performance in the simulation results. In this paper, we will propose fast approximate algorithms with the controllable performance bound and the running time based on the work in [9] with some modifications necessary for our problem.

In our study the topology of the network is static and the routing accounts to finding the traffic splits that balance optimally the energy consumption. Hence the results are applicable to networks which are either static, like the sensor networks we mentioned earlier, or their topology changes slowly enough such that there is enough time for optimally balancing the traffic in the periods between successive topology changes.

This paper is organized as follows: In Section 2, the problem is formulated. In Section 3, fast approximate algorithms are proposed in the single commodity case and in the multicommodity case. In Section 4, the problem is extended to the case where there is an additional constraint on the delay.

## 2 Flows, Optimal Energy Consumption and Transmission Power Levels

In this section, the problem formulation given in our previous work[3] is repeated for the sake of completeness of the presentation.

The wireless ad-hoc network in consideration is modeled as a directed graph  $G(N, A)$  where  $N$  is the set of all nodes and  $A$  is the set of all directed links  $(i, j)$  where  $i, j \in N$ . Let  $S_i$  be the set of all nodes that can be reached by node  $i$  with a certain power level in its dynamic range. We assume that link  $(i, j)$  exists if and only if  $j \in S_i$ . Let each node  $i$  have the initial battery energy  $E_i$ , and let  $Q_i^{(c)}$  be the rate at which information is generated at node  $i$  belonging to commodity  $c \in C$ , where  $C$  is the set of all commodities. We are given, for each commodity  $c$ , an origin node  $o^{(c)}$  where the information is generated, and a destination node  $d^{(c)}$ . Assume that the transmission energy required for node  $i$  to transmit an information unit to its neighboring node  $j$  is  $e_{ij}$ , and the rate at which information of commodity  $c$  is transmitted from node  $i$  to node  $j$  is called the flow  $q_{ij}^{(c)}$ .

The lifetime of node  $i$  under a given flow  $\mathbf{q} = \{q_{ij}^{(c)}\}$  is given by

$$T_i(\mathbf{q}) = \frac{E_i}{\sum_{j \in S_i} e_{ij} \sum_{c \in C} q_{ij}^{(c)}}. \quad (1)$$

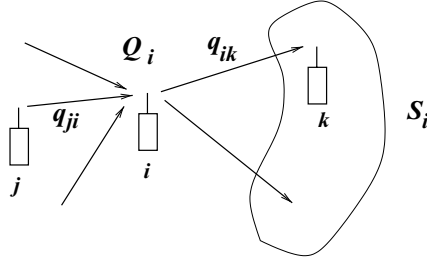
Now, let us define the *system lifetime* under flow  $\mathbf{q}$  as the minimum lifetime over all nodes, i.e.,

$$T_{sys}(\mathbf{q}) = \min_{i \in N} T_i(\mathbf{q}). \quad (2)$$

Our goal is to find the flow that maximizes the system lifetime under the flow conservation condition drawn in Fig.2, and the problem can be written as follows:

$$\begin{aligned} & \text{Maximize} && T_{sys}(\mathbf{q}) \\ & \text{s.t.} && q_{ij}^{(c)} \geq 0, && \forall i \in N, \forall j \in S_i, \forall c \in C, \\ & && \sum_{j: i \in S_j} q_{ji}^{(c)} + Q_i^{(c)} = \sum_{k \in S_i} q_{ik}^{(c)}, && \forall c \in C, \forall i \in N - \{d^{(c)}\}. \end{aligned} \quad (3)$$

The problem of maximizing the system lifetime, given the information generation rates  $Q_i^{(c)}$  at the origin node  $o^{(c)}$  for each commodity  $c$ , can be rewritten



**Fig. 2.** The conservation of flow condition at node  $i$  is illustrated.

as the following linear programming problem:

$$\begin{aligned}
 & \text{Maximize} && T \\
 & \text{s.t.} && \hat{q}_{ij}^{(c)} \geq 0, && \forall i \in N, \forall j \in S_i, \forall c \in C, \\
 & && \sum_{j \in S_i} e_{ij} \sum_{c \in C} \hat{q}_{ij}^{(c)} \leq E_i, && \forall i \in N, \\
 & && \sum_{j: i \in S_j} \hat{q}_{ji}^{(c)} + TQ_i^{(c)} = \sum_{k \in S_i} \hat{q}_{ik}^{(c)}, && \forall c \in C, \forall i \in N - \{d^{(c)}\},
 \end{aligned} \tag{4}$$

where  $\hat{q}_{ij}^{(c)} = Tq_{ij}^{(c)}$  is the amount of information of commodity  $c$  transmitted from node  $i$  to node  $j$  until time  $T$ . Note that when the information generation rate is fixed, the objective of maximizing the system lifetime is equivalent to maximizing the amount of data transfer.

The linear program given above can be viewed as a variation of the conventional maximum flow problem with node capacities[5]. If the transmitted power level at each node is fixed regardless of its next hop node, i.e., if there is no power control, then

$$e_{ij} = e_i, \quad \forall j \in S_i, \tag{5}$$

and the problem is equivalent to the maximum flow problem with node capacities given by

$$\sum_{j \in S_i} \sum_{c \in C} \hat{q}_{ij}^{(c)} \leq E_i/e_i, \quad \forall i \in N. \tag{6}$$

When the capacity of a node is a fixed quantity as in (6) then the problem can be converted to a link capacity version by replacing the node with two nodes and a link having the same capacity[4], and the max-flow-min-cut theorem[5] can be used. However, in our problem, unlike the above, the amount of resource (or energy in this case) which a unit flow consumes depends on the energy expenditure to the next hop node. Therefore, it is not trivial to find the min-cut nodes, and even if they were found the traffic split at the nodes must also be identified.



### 3 Fast Approximate Algorithms

In this section, fast approximate algorithms will be proposed in single commodity case and in multicommodity case along with the analysis which enables us to control the performance and the complexity of the algorithms.

#### 3.1 Single Commodity Case

Let  $o$  and  $d$  denote the origin node and the destination node, respectively. Let  $P$  be the set of all paths between  $o$  and  $d$ , and let  $f_i$  be the value of the flow on the  $i$ th path in  $P$ . The problem in (4), in the single commodity case, can be rewritten as follows:

$$\begin{aligned} & \text{Maximize} && \sum_{i \in P} f_i \\ & \text{s.t.} && f_i \geq 0, \quad \forall i \in P, \\ & && \sum_{j \in \mathcal{P}_i} e_{l(i,j)} f_j \leq E_i, \quad \forall i \in N - \{d\}, \end{aligned} \quad (7)$$

where  $\mathcal{P}_i$  is the set of all paths that passes node  $i$ , and  $l(i, j)$  is the link on path  $j$  that stems from node  $i$ .

The problem above is a linear program and its dual problem is given by

$$\begin{aligned} & \text{Minimize} && \sum_{i \in N} \lambda_i E_i \\ & \text{s.t.} && \lambda_i \geq 0, \quad \forall i \in N, \\ & && \sum_{j \in \mathcal{N}_i} \lambda_j e_{l(j,i)} \geq 1, \quad \forall i \in P, \end{aligned} \quad (8)$$

where  $\mathcal{N}_i$  is the set of all nodes on path  $i$  except the destination node  $d$ .

Physical interpretation of the dual problem is the assignment of node weights  $\lambda_i$  such that  $\sum_{i \in N} \lambda_i E_i$  is minimized with the constraint that the length of the shortest length path is at least one, where the length of a path is defined as the weighted sum of energy expenditure per unit flow over each link in the path by the weight of its head node.

The proposed algorithm is described in the following.

Let  $J(\lambda(x))$  be defined as the objective function at the  $x$ th iteration, i.e.,

$$J(\lambda(x)) = \sum_{i \in N} \lambda_i(x) E_i,$$

and let  $\alpha(\lambda(x))$  be defined as the length of the shortest path  $sp(x)$  at the  $x$ th iteration, i.e.,

$$\begin{aligned} \alpha(\lambda(x)) &= \min_{j \in P} \sum_{i \in \mathcal{N}_j} \lambda_i(x) e_{l(i,j)} \\ &= \sum_{i \in \mathcal{N}_{sp(x)}} \lambda_i(x) e_{l(i, sp(x))}. \end{aligned}$$

The problem is equivalent to minimizing  $\frac{J(\lambda(x))}{\alpha(\lambda(x))}$ , and let  $\beta$  be this minimum, i.e.,

$$\beta = \min_{\lambda(x)} \frac{J(\lambda(x))}{\alpha(\lambda(x))}.$$

Let each node  $i$  have initial weight given by

$$\lambda_i(0) = \frac{\delta}{E_i}, \quad (9)$$

where  $\delta$  is a constant which will be determined later. At the  $x$ th iteration, an amount of  $\Delta f(x)$  of flow is augmented on the shortest path  $sp(x-1)$ , where  $\Delta f(x)$  is given by

$$\Delta f(x) = \min_{i \in \mathcal{N}_{sp(x-1)}} \frac{E_i}{e_{l(i, sp(x-1))}}, \quad (10)$$

which is the capacity of the minimum capacity link on the shortest path  $sp(x-1)$ . If we denote the total flow assigned until the  $x$ th iteration by  $f(x)$  then  $f(0) = 0$  and  $f(x) = f(x-1) + \Delta f(x)$ . After the augmentation, the node weights on the shortest path are updated by

$$\lambda_i(x) = \lambda_i(x-1) \{1 + \epsilon \Delta f(x) e_{l(i, sp(x-1))} / E_i\}, \quad (11)$$

for all  $i \in \mathcal{N}_{sp(x-1)}$ , where  $\epsilon$  is a constant which will be determined later. Note that  $\lambda_i(x) = \lambda_i(x-1)$  for all the other nodes that are not on the shortest path. Using the updated  $\lambda_i(x)$ , shortest path  $sp(x)$  is calculated. The shortest path calculation and the flow augmentation are repeated until iteration  $t$  where  $J(\lambda(t)) \geq 1$  for the first time.

In the following, we analyze the algorithm's performance and its running time.

For all  $x \geq 1$ ,

$$\begin{aligned} J(\lambda(x)) &= \sum_{i \in N} \lambda_i(x-1) E_i + \epsilon \Delta f(x) \sum_{i \in \mathcal{N}_{sp(x-1)}} \lambda_i(x-1) e_{l(i, sp(x-1))} \\ &= J(\lambda(x-1)) + \epsilon \{f(x) - f(x-1)\} \alpha(\lambda(x-1)). \end{aligned}$$

Since

$$\beta \leq \frac{J(\lambda(x-1))}{\alpha(\lambda(x-1))},$$

it follows that

$$\begin{aligned} J(\lambda(x)) &\leq J(\lambda(x-1)) \left[1 + \frac{\epsilon}{\beta} \{f(x) - f(x-1)\}\right] \\ &\leq J(\lambda(x-1)) \exp\left(\frac{\epsilon}{\beta} \{f(x) - f(x-1)\}\right), \end{aligned}$$

which implies that

$$J(\lambda(x)) \leq n\delta \exp\left(\frac{\epsilon}{\beta} f(x)\right).$$

By our stopping condition,

$$1 \leq J(\lambda(t)) \leq n\delta \exp\left(\frac{\epsilon}{\beta} f(t)\right).$$

Therefore,

$$\frac{\beta}{f(t)} \leq \frac{\epsilon}{\ln(n\delta)^{-1}}. \quad (12)$$

There is a feasible flow of value  $\frac{f(t)}{\log_{1+\epsilon} \frac{1+\epsilon}{\delta}}$ , which can be shown as follows. Consider node  $i$ . For every  $E_i$  consumed, the weight increases by a factor of at least  $1 + \epsilon$ . Before the algorithm stops,  $J(\lambda(t-1)) = \sum_{i \in N} \lambda_i(t-1)E_i < 1$ , which means that  $\lambda_i(t-1) < 1/E_i$  for each  $i$ . At each iteration the node weight is increased by a factor of at most  $(1 + \epsilon)$ , so  $\lambda_i(t) < (1 + \epsilon)/E_i$ . Initially, the node weight was  $\delta/E_i$ , and  $E_i$  may have been consumed at most  $t$  times. Therefore,

$$\frac{\delta}{E_i} (1 + \epsilon)^t < \frac{(1 + \epsilon)}{E_i},$$

and the total consumed energy at node  $i$  is at most

$$E_i \log_{1+\epsilon} \frac{1 + \epsilon}{\delta}.$$

The final flow  $f(t)$  may have violated the energy constraint, and can be made feasible by dividing it by  $\log_{1+\epsilon} \frac{1+\epsilon}{\delta}$ . Note that the resulting flow from this algorithm represents the amount of possible information transfer. To obtain the system lifetime, it should be divided by the information generation rate.

Now, the ratio between the optimum dual and the primal solutions is given by

$$\gamma = \frac{\beta}{f(t)} \log_{1+\epsilon} \frac{1 + \epsilon}{\delta}.$$

Substituting this to (12) gives

$$\begin{aligned} \gamma &\leq \frac{\epsilon \log_{1+\epsilon} \frac{1+\epsilon}{\delta}}{\ln(n\delta)^{-1}} \\ &= \frac{\epsilon}{\ln(1 + \epsilon)} \frac{\ln \frac{1+\epsilon}{\delta}}{\ln(n\delta)^{-1}}. \end{aligned}$$

The ratio  $\frac{\ln \frac{1+\epsilon}{\delta}}{\ln(n\delta)^{-1}}$  equals  $(1 - \epsilon)^{-1}$  when

$$\delta = (1 + \epsilon) \{(1 + \epsilon)n\}^{-1/\epsilon}. \quad (13)$$

Hence with this choice of  $\delta$  we have

$$\begin{aligned} \gamma &\leq \frac{\epsilon}{(1 - \epsilon) \ln(1 + \epsilon)} \\ &\leq \frac{\epsilon}{(1 - \epsilon)(\epsilon - \epsilon^2/2)} \\ &\leq \frac{1}{(1 - \epsilon)^2}. \end{aligned}$$

Therefore,  $\epsilon$  is chosen to be

$$\epsilon = 1 - \frac{1}{\sqrt{\gamma}}, \quad (14)$$

in order to guarantee the  $\gamma$ -approximation.

The total number of iterations in which a node is the head of the minimum capacity link on the path chosen in that iteration is at most

$$\lceil \log_{1+\epsilon} \frac{1}{\delta} \rceil = \lceil \frac{1}{\epsilon} - 1 + \frac{1}{\epsilon} \log_{1+\epsilon} n \rceil.$$

Using the fact that there are  $n$  nodes we get the following running time. The running time of the algorithm that computes a  $(1 - \epsilon)^{-2}$ -approximation to the optimal solution is at most

$$n \lceil \frac{1}{\epsilon} - 1 + \frac{1}{\epsilon} \log_{1+\epsilon} n \rceil T_{sp},$$

which is  $O(n(\frac{1}{\epsilon} \log_{1+\epsilon} n) T_{sp})$ , where  $T_{sp}$  is the time required to compute the shortest path in a graph with non-negative link lengths.

### 3.2 Multicommodity Case

Assume that each commodity has a single origin and destination pair. Note that the case of having multiple destinations where any one of them needs to be reached can be obtained by inserting an imaginary node connecting all destinations, and the case of having multiple origins can be obtained by treating each as a separate commodity. Let  $Q^{(c)}$  be the information generation rate of commodity  $c$  at its origin node  $o^{(c)}$ . Let  $\hat{P}$  be the set of all path clusters where a path cluster consists of one path for each commodity. Let  $f_i$  be such that the flow value of commodity  $c$  on the path cluster  $i$  is  $Q^{(c)} f_i$ . Note that we are maximizing the flow while keeping the ratios among the information generation rates of all commodities. The problem in (4) can be rewritten as follows:

$$\begin{aligned} & \text{Maximize} && \sum_{i \in \hat{P}} f_i \\ & \text{s.t.} && f_i \geq 0, \quad \forall i \in \hat{P}, \\ & && \sum_{c \in C} \sum_{j \in \mathcal{P}_i^{(c)}} e_{l^{(c)}(i,j)} Q^{(c)} f_j \leq E_i, \quad \forall i \in N, \end{aligned} \quad (15)$$

where  $\mathcal{P}_i^{(c)}$  is the set of all path clusters whose path for commodity  $c$  passes node  $i$ , and  $l^{(c)}(i,j)$  is the link that commodity  $c$  of path cluster  $j$  visits right after visiting node  $i$ .

The dual problem is given by

$$\begin{aligned} & \text{Minimize} && \sum_{i \in N} \lambda_i E_i \\ & \text{s.t.} && \lambda_i \geq 0, \quad \forall i \in N, \\ & && \sum_{c \in C} Q^{(c)} \sum_{j \in \mathcal{N}_i^{(c)}} \lambda_j e_{l^{(c)}(j,i)} \geq 1, \quad \forall i \in \hat{P}, \end{aligned} \quad (16)$$

where  $\mathcal{N}_i^{(c)}$  is the set of all nodes on the path for commodity  $c$  of path cluster  $i$ .

The algorithm is described in the following.

Let  $J(\lambda(x))$  be defined as

$$J(\lambda(x)) = \sum_{i \in N} \lambda_i(x) E_i,$$

and let  $\alpha(\lambda(x))$  be defined as

$$\alpha(\lambda(x)) = \frac{1}{m} \sum_{c \in C} Q^{(c)} \sum_{i \in \mathcal{N}_{sp^{(c)}(x)}^{(c)}} \lambda_i(x) e_{l(i, sp^{(c)}(x))},$$

where  $sp^{(c)}(x)$  is the shortest path from  $o^{(c)}$  to  $d^{(c)}$  at the  $x$ th iteration, and  $m$  is the number of commodities.

Our objective is to minimize  $J(\lambda(x))/\alpha(\lambda(x))$ . Let  $\beta$  be this minimum, i.e.,

$$\beta = \min_{\lambda(x)} \frac{J(\lambda(x))}{\alpha(\lambda(x))}.$$

The node weights are initialized as

$$\lambda_i(0) = \frac{\delta}{E_i}, \quad (17)$$

for all  $i \in N$  as before. At each iteration  $x$ , for each commodity  $c$ , the shortest length path to the destination  $d^{(c)}$  is at hand from the previous iteration, which is denoted by  $sp^{(c)}(x-1)$ . For each commodity  $c$ , an amount of  $Q^{(c)} \Delta f(x)$  is augmented on its shortest length path, where  $\Delta f(x)$  is calculated by

$$\Delta f(x) = \min_{c \in C} \min_{i \in \mathcal{N}_{sp^{(c)}(x-1)}^{(c)}} \frac{E_i}{e_{l(i, sp^{(c)}(x-1))}} \frac{1}{Q^{(c)}}, \quad (18)$$

and the total flow assigned until iteration  $x$  is updated by

$$f(x) = f(x-1) + \Delta f(x).$$

The node weights on the shortest path are updated as follows.

$$\lambda_i(x) = \lambda_i(x-1) \left\{ 1 + \frac{\epsilon}{m} \sum_{c \in C} \frac{Q^{(c)} \Delta f(x)}{E_i / e_{l(i, sp^{(c)}(x-1))}} \right\}, \quad (19)$$

for all  $i \in \mathcal{N}_{sp^{(c)}(x-1)}$ , where  $\epsilon$  is a constant to be chosen later. Finally, the algorithm stops after  $t$  iterations, where  $J(\lambda(t)) \geq 1$  for the first time.

There is a feasible flow of value  $\frac{f(t)}{\log_{1+\epsilon} \frac{1+\epsilon}{\delta}}$ , which can be shown as in the single commodity case. The analysis is the same as that in the single commodity case except for the running time. We have  $n$  nodes, and for each iteration, we need  $m$  shortest path calculations. Therefore, the running time is

$$n \left\lceil \frac{1}{\epsilon} - 1 + \frac{1}{\epsilon} \log_{1+\epsilon} n \right\rceil m T_{sp}, \quad (20)$$

which is  $O(n(\frac{1}{\epsilon} \log_{1+\epsilon} n) m T_{sp})$ .

## 4 Delay Constrained Maximum Lifetime Problem

In this section, we consider the problem with an additional constraint on delay. For simplicity, the single commodity case will be treated only. Extension to the multicommodity case will be straightforward.

The problem is defined as maximizing the system lifetime with the constraint that the delay between the origin node  $o$  and the destination node  $d$  be less than or equal to  $H$ , which can be written as

$$\begin{aligned}
 & \text{Maximize} && T \\
 & \text{s.t.} && \hat{q}_{ij} \geq 0, && \forall i \in N - \{d\}, \forall j \in S_i, \\
 & && \sum_{j \in S_i} e_{ij} \hat{q}_{ij} \leq E_i, && \forall i \in N - \{d\}, \\
 & && \sum_{j : i \in S_j} \hat{q}_{ji} + TQ_i = \sum_{k \in S_i} \hat{q}_{ik}, && \forall i \in N - \{d\}, \\
 & && \sum_{j \in \mathcal{N}_i} \tau_{l(j,i)} \leq H, && \forall i \in \{i \mid \hat{q}_{l(j,i)} > 0, \forall j \in \mathcal{N}_i\},
 \end{aligned} \tag{21}$$

where  $\tau_{l(j,i)}$  is the delay from node  $j$  to its next hop on path  $i$ .

The problem above can be rewritten using the flow variables as before, and the primal problem is given by

$$\begin{aligned}
 & \text{Maximize} && \sum_{i \in P} f_i \\
 & \text{s.t.} && f_i \geq 0, && \forall i \in P, \\
 & && \sum_{j \in P_i} e_{l(i,j)} f_j \leq E_i, && \forall i \in N - \{d\}, \\
 & && \sum_{j \in \mathcal{N}_i} \tau_{l(j,i)} f_i \leq H f_i, && \forall i \in P.
 \end{aligned} \tag{22}$$

The dual problem of the above is given by

$$\begin{aligned}
 & \text{Minimize} && \sum_{i \in N} \lambda_i E_i \\
 & \text{s.t.} && \lambda_i \geq 0, && \forall i \in N, \\
 & && \eta_i \geq 0, && \forall i \in P, \\
 & && \sum_{j \in \mathcal{N}_i} \lambda_j e_{l(j,i)} + \eta_i \left( \sum_{j \in \mathcal{N}_i} \tau_{l(j,i)} - H \right) \geq 1, && \forall i \in P.
 \end{aligned}$$

We propose an algorithm which applies the same idea as in section 3.1. However, instead of incrementing the flow on the shortest path we use the delay constrained shortest path which is the shortest path of all  $i \in P$  that satisfies  $\sum_{j \in \mathcal{N}_i} \tau_{l(j,i)} - H \leq 0$ . In other words, we are enforcing the delay constraints to be met in each iteration, and setting  $\eta_i = 0$  for all  $i$  would result in a larger feasible region for  $\lambda_i$ 's, which in turn will yield a better optimization.

The description of the proposed algorithm is exactly the same as before except that the shortest path is replaced by the delay constrained shortest path.

The analysis of the performance and the running time is exactly the same as that in section 3.1 without the delay constraint. The one and most important

difference is that the problem of finding a delay constrained shortest path is NP-complete. The decision version of the problem is shown to be NP-complete in [1].

If we assume that the delay between any two nodes is fixed, say  $\tau$ , then the delay constraint becomes the constraint on the number of hops which must be less than or equal to  $\lfloor \frac{H}{\tau} \rfloor$ . Since the shortest path obtained after  $x$  iterations in the Bellman-Ford algorithm provides the shortest path with at most  $x$  hops, the delay constrained shortest path problem in this case becomes solvable in polynomial time by limiting the number of iteration in the Bellman-Ford algorithm by that many iterations instead of  $N - 1$  iterations.

## 5 Conclusion

In this paper, we have proposed fast approximate algorithms for the maximum lifetime routing problem in wireless ad-hoc networks in the single commodity case and in the multicommodity case with the analysis on the performance bound and the running time. In addition, an extended problem where there is an additional delay constraint has been studied. The delay constrained shortest path problem is NP-complete in general. However, the same algorithm can be used in the special case where the delay constraint is given as the maximum number of hops from the origin node to destination node by limiting the number of iterations in the Bellman-Ford algorithm.

## References

1. Ravindra K. Ahuja, Thomas L. Magnanti, and James B. Orlin. *Network Flows : Theory, Algorithms, and Applications*. Prentice Hall, 1993.
2. Dennis J. Baker and Anthony Ephremides. The architectural organization of a mobile radio network via a distributed algorithm. *IEEE Transactions on Communications*, COM-29(11):56–73, January 1981.
3. Jae-Hwan Chang and Leandros Tassiulas. Energy conserving routing in wireless ad-hoc networks. In *Proceedings of IEEE INFOCOM2000*, Tel Aviv, Israel, March 2000.
4. Wai-Kai Chen. *Theory of Nets: Flows in Networks*. Wiley, 1990.
5. T. Cormen, C. Leiserson, and R. Rivest. *Introduction to Algorithms*. McGraw-Hill and MIT Press, 1990.
6. Anthony Ephremides, Jeffrey E. Wieselthier, and Dennis J. Baker. A design concept for reliable mobile radio networks with frequency hopping signaling. *Proceedings of the IEEE*, 75(1):56–73, January 1987.
7. M. Ettus. System capacity, latency, and power consumption in multihop-routed SS-CDMA wireless networks. In *Proceedings of IEEE Radio and Wireless Conference (RAWCON) 98*, pages 55–58, Colorado Springs, CO, August 1998.
8. R.G. Gallager, P.A. Humblet, and P.M. Spira. A distributed algorithm for minimum weight spanning trees. Technical Report LIDS-P-906-A, Lab. Inform. Decision Syst., Massachusetts Inst. of Technol., Cambridge, MA, October 1979.

9. Naveen Garg and Jochen Koenemann. Faster and simpler algorithms for multicommodity flow and other fractional packing problems. In *Proceedings 39th Annual Symposium on Foundations of computer science*, pages 300–309, Palo Alto, CA, November 1998.
10. D. Johnson and D. Maltz. Dynamic source routing in ad hoc wireless networks. In Tomasz Imielinski and Hank Korth, editors, *Mobile Computing*, chapter 5, pages 153–181. Kluwer Academic Publishers, 1996.
11. Teresa H. Meng and Volkan Rodoplu. Distributed network protocols for wireless communication. In *Proceedings of the 1998 IEEE International Symposium on Circuits and Systems, ISCAS'98*, volume 4, pages 600–603, Monterey, CA, June 1998.
12. A. Michail and A. Ephremides. A distributed routing algorithm for supporting connection-oriented service in wireless networks with time-varying connectivity. In *Proceedings Third IEEE Symposium on Computers and Communications, ISCC'98*, pages 587–591, Athens, Greece, June 1998.
13. S. Murthy and J.J. Garcia-Luna-Aceves. An efficient routing protocol for wireless networks. *Journal of Special Topics in Mobile Networks and Applications (MONET)*, 1(2):183–197, 1996.
14. Vincent D. Park and M. Scott Corson. A highly distributed routing algorithm for mobile wireless networks. In *Proc. IEEE INFOCOM'97*, pages 1405–1413, Kobe, Japan, 1997.
15. C. Perkins and P. Bhagwat. Highly dynamic destination-sequenced distance vector routing (DSDV) for mobile computers. In *ACM SIGCOMM*, pages 234–244, London, England, August 1994.
16. Volkan Rodoplu and Teresa H. Meng. Minimum energy mobile wireless networks. In *Proceedings of the 1998 IEEE International Conference on Communications, ICC'98*, volume 3, pages 1633–1639, Atlanta, GA, June 1998.
17. Timothy Shepard. Decentralized channel management in scalable multihop spread spectrum packet radio networks. Technical Report MIT/LCS/TR-670, Massachusetts Institute of Technology Laboratory for Computer Science, July 1995.
18. S. Singh, M. Woo, and C.S. Raghavendra. Power-aware routing in mobile ad hoc networks. In *Proceedings of Fourth Annual ACM/IEEE International Conference on Mobile Computing and Networking*, pages 181–190, Dallas, TX, October 1998.



# Performance Evaluation of Resource Division Policies for PGPS Scheduling in ATM Networks

Amr S. Ayad, Khaled M. F. Elsayed, and Mahmoud T. El-Hadidi

Department of Electronics and Communications Engineering  
Faculty of Engineering, Cairo University, Giza, Egypt 12613  
asaad@ie-eg.com, khaled@ieee.org, hadidi@frcu.eun.eg

**Abstract.** The paper addresses the issue of reserving resources at ATM switches along the path of calls requiring a deterministic bound on end-to-end delay. The switches are assumed to schedule outgoing packets using the Packet-by-Packet Generalized Processor Sharing (PGPS) scheduling discipline. We propose an algorithm for call admission control (CAC), and present the simulation results used to evaluate the performance of the resource division policies used for mapping the end-to-end delay requirement into service rates to be reserved at each switch. The simulation results also show the performance gain when a simple resource-based routing algorithm is used.

## 1 Introduction

One of the main promises of ATM networks is to provide users with Quality-of-Service (QoS) guarantees, such as Cell Transfer Delay (CTD) and Cell Loss Ratio (CLR). Handling the variety in QoS requirements of different applications requires the network to use a mechanism for serving packets from different applications according to their contracted QoS level. Many packet-scheduling disciplines have been proposed in the literature to implement such mechanisms (see [4], [8], [9]). Each scheduling discipline requires algorithms for performing call admission control (CAC) and resource reservation. In this paper, we propose such algorithms for the case of PGPS service discipline and calls requiring a hard (deterministic) bound on end-to-end delay. The paper addresses the following problems:

1. How to map the end-to-end delay requirement of a call into a local resource requirement to be reserved at each switch along the call's path?
2. How to divide the resource requirement among the schedulers on the call's path? That is, a simple even division policy would be to reserve the same amount of resources at all schedulers. However, it may be more efficient to use a policy that takes schedulers capacities and/or loading into account.
3. How much gain (if any) would be obtained from applying non-even resource division policies? What are the factors controlling the gain value?

The following terms will be used throughout the paper:

1. *Resource load on a switch*: It represents the amount of reserved resources at a switch to satisfy the guaranteed QoS of accepted calls. This value depends on the calls' traffic characteristics and the QoS level requested by each call. For PGPS, this is expressed in bit rate units (e.g. bps)
2. *Call load*: It represents the arrival rate and the holding time of incoming calls without regard to their resource load. The call load is measured in Erlangs.

The rest of this paper is organized as follows: section 2 gives an overview of the delay formulas associated with PGPS scheduling. Section 3 presents the proposed CAC algorithm, and the associated non-even resource division policies. Section 4 presents the simulation results of applying the proposed algorithms to several network models. Section 5 presents the simulation results showing the performance enhancement resulting from the use of resource-based routing. Section 6 concludes the paper.

## 2 PGPS Scheduling Discipline

### 2.1 Delay Formulas

Generalized Processor Sharing (GPS) (see [6]) is an ideal and non-realizable scheduling discipline that serves packets as if they are in separate logical queues, visiting each nonempty queue in turn and serving an *infinitesimally* small amount of data from each queue. Connections can be associated with service weights, and they receive service in proportion to this weight whenever they have data in the queue. PGPS scheduling (see [8]) closely approximates GPS by serving packets in ascending order of the service tags assigned to them. The service tags are computed as the finish times of those packets had a GPS scheduler having the same capacity and the same input traffic served them.

This paper builds on the work done in [1], [5], and [8] where it was shown that if a call ( $f$ ) traverses a path of  $K_f$  PGPS schedulers and has traffic characteristics conforming to a leaky bucket with a maximum burst size of  $(\sigma_f)$  bits and a long term average rate of  $(\rho_f)$  bps, then an upper bound on the end-to-end delay is guaranteed for each ATM cell from call ( $f$ ) by reserving a certain service rate at each switch along the call's path. The upper bound ( $D_f$ ) is given by:

$$D_f \leq \frac{\sigma_f - L}{g_f} + \left( \sum_{j=1}^{j=K_f} \alpha^j \right) + \left( \sum_{j=1}^{j=K_f} \frac{L}{g_f^j} \right). \quad (1)$$

where:

$g_f^j$  = The service rate reserved for call ( $f$ ) at switch ( $j$ ),  $g_f = \min_{j \in [1, K_f]} g_f^j$ ,

$\alpha^j = \beta^j + \tau^{j, j+1}$ ,  $\beta^j = L/C^j$ ,  $L$  = Length of the ATM cell (424 bits).

$\tau^{j, j+1}$  = The propagation delay on the link from switch ( $j$ ) to switch ( $j+1$ )

$C^j$  = The data rate of the link following switch ( $j$ ) in bps. We will denote  $C^j$  as the switch capacity.

The above inequality is only valid when the following conditions are met at each switch ( $j$ ),  $j \in [1, K_j]$ .

1-*The scheduler stability condition*, which requires that:

$$\sum_{k=1}^N \rho_k \leq C^j. \quad (2)$$

where  $N$  is the number of accepted calls at switch ( $j$ ).

The stability condition is necessary for all scheduling disciplines and is not specific to PGPS scheduling.

2-*The schedulability condition* for PGPS schedulers, which requires that:

$$\sum_{k=1}^N g_k^j \leq C^j. \quad (3)$$

where  $N$  is the number of accepted calls at switch ( $j$ ).

## 2.2 The Condition of Local Stability

A call ( $f$ ) is said to be locally stable at a switch ( $j$ ) if

$$g_f^j \geq \rho_f. \quad (4)$$

The local stability condition is not required for each call passing by a given switch if all the calls passing by this switch have a leaky-bucket constrained traffic. However, if some call is not, then (4) must hold true for all accepted calls. Therefore, the local stability condition is not necessary if the network operator uses leaky bucket traffic shapers for all the network's ingress traffic.

The implementation of the proposed resource division algorithms depends on whether local stability is imposed or not. For shortness, we will only consider the case in which all traffic is leaky-bucket shaped and, thus, the local stability condition need not be imposed when reserving rates for new calls.

The other case where this condition must be applied to all calls requires a modification of the algorithms presented in this paper and has been addressed in [2].

## 3 CAC Algorithm and Resource Division Policies

### 3.1 CAC Algorithm

The proposed CAC algorithm uses equations (1)-(3) to determine whether to accept or reject a new call. It operates as follows:

The first test compares the value of the end-to-end delay ( $D_f$ ) requested by the incoming call with the value of the total transmission and propagation delay along the call's path. If the value of the required end-to-end delay is smaller, the call is rejected. The next test is to verify that the value of the required end-to-end delay of the incoming call is not less than the minimum end-to-end delay bound that the network can guarantee to the incoming call. This minimum value ( $D_f^*$ ) is obtained from (1)

with each switch along the call's path reserving a service rate equal to its remaining capacity. We define the remaining capacity of a PGPS scheduler ( $j$ ) prior to the acceptance of call ( $f$ ) as:

$$R_f^j = C^j - \sum_{k=1}^{N_f} g_k^j . \quad (5)$$

where  $N_f$  is the number of accepted calls prior to the acceptance of call ( $f$ ). We denote the minimum remaining capacity along the path of call ( $f$ ) prior to accepting it by  $R_f$ .

On passing the previous tests successfully, a division policy is used to map the required end-to-end delay into a local resource requirement  $\{g_f^j\}$  to be reserved at each switch. Different division policies are discussed in the next section.

Finally a test is made to verify that the conditions in (2) and (3) hold true for all schedulers on the call path. If local stability is imposed, then the local stability condition in (4) must also hold true. If all conditions are met, the call is accepted

### 3.2 Resource Division Policies

**Even policy (EVEN).** The reserved rates are the same at all schedulers, i.e.

$$g_f^i = g_f \forall i \in [1, K_f] . \quad (6)$$

Substituting in (1), after converting it to an equality to reserve the least amount of resources required for meeting the delay bound of call ( $f$ ), we get:

$$g_f^i = g_f = \frac{\sigma_f + (K_f - 1)L}{D_f - \sum_{j=1}^{K_f} \alpha^j} \forall i \in [1, K_f] . \quad (7)$$

**Capacity proportional policy (CP).** The reserved rate at a certain switch is proportional to the switch capacity, i.e.

$$g_f^i = \eta_f C^i \forall i \in [1, K_f] . \quad (8)$$

where  $\eta_f = \text{Constant}$  for the path and call ( $f$ ) parameters.

Substituting in (1), after converting it to an equality to reserve the least amount of resources required to meet the delay bound and solving for  $\eta_f$ , we get:

$$g_f^i = \frac{\frac{\sigma_f - L}{C} + \sum_{j=1}^{K_f} (L/C^j)}{D_f - \sum_{j=1}^{K_f} \alpha^j} C^i \forall i \in [1, K_f] . \quad (9)$$

where  $C = \min_j C^j$ .

**Remaining capacity proportional policy (RCP).** The reserved rate at a certain switch is proportional to the remaining capacity of the switch i.e.

$$g_f^i = \eta_f R_f^i \forall i \in [1, K_f] . \quad (10)$$

where  $\eta_f = \text{Constant}$  for the path and call ( $f$ ) parameters as long as no other calls are accepted at any of the schedulers along the call path's during call setup phase. Substituting in (1) after converting it to an equality to reserve the least amount of resources required for meeting the delay bound and solving for  $\eta_f$ , we get:

$$g_f^i = \frac{\frac{\sigma_f - L}{R_f} + \sum_{j=1}^{K_f} (L/R_f^j)}{D_f - \sum_{j=1}^{K_f} \alpha^j} R_f^i \forall i \in [1, K_f] . \quad (11)$$

Note that computing the rate to be reserved at each switch using the above division policies may result in a case in which the rate computed from equations (7) or (9) is greater than the remaining capacity at one or more schedulers, i.e.  $g_f^n \geq R_f^n$  for some  $n \in [1, K_f]$ . We denote such schedulers as *resource-limited schedulers*.

It can be shown [1] that using the RCP policy in conjunction with the proposed CAC algorithm guarantees the absence of resource-limited schedulers when accepting a new call. Thus, this case is only present with EVEN and CP policies. There are two approaches for handling the existence of resource-limited schedulers on accepting a new call:

- a. Use an algorithm that reserves all of the remaining capacity (i.e.  $g_f^i = R_f^i$ ) at such schedulers and then redistributes the rest of the delay requirement on other schedulers using (7), or (9).
- b. Reject the incoming call.

The first approach seems to be more efficient. However, it requires more state information to be exchanged among the switches and also requires more computations to be made by the CAC algorithm. The presented simulations results are mainly based on the second approach (simply rejecting the new call). Reference [2] presents the simulation results when using the first approach.

## 4 Simulation Results

We have simulated the operation of the proposed CAC algorithm and the associated resource-division policies on several network models. The simulation consisted of generating a number of calls according to a Poission distribution with an average arrival rate of  $\lambda$ , and a holding time that is exponentially distributed with a mean of  $1/\mu$ . The value  $\rho = \lambda/\mu$  characterizes the call load offered to the model.

The main objective is to compare the blocking probabilities of different division policies. An estimate of the blocking probability is computed as the number of blocked calls divided by the total number of generated calls. We simulated the following configurations of link capacities for each network model:

1-*Configuration (A)*: In this configuration, all links have the same capacity. Therefore, the results for the EVEN and CP policies are always the same.

*2-Configuration (B):* In this configuration, link capacities are chosen in proportion to the expected call load.

*3-Configuration (C):* In this configuration, link capacities are chosen in inverse proportion to the expected call load. It may be argued that this assignment of capacities is not typical for a properly planned network. However, we argue that there are two reasons leading to the importance of studying such configuration:

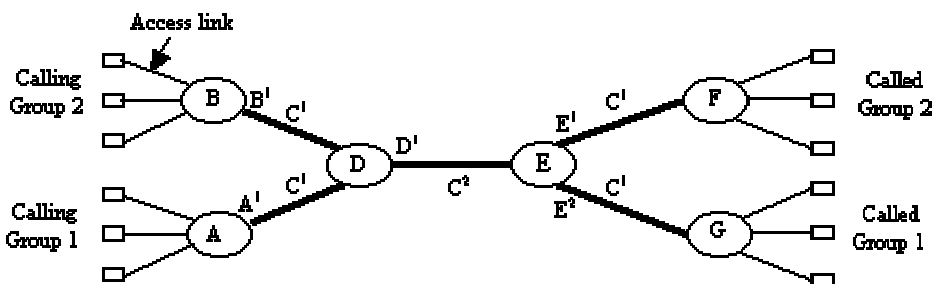
- It may be difficult at the time of initial network planning to determine the actual load distribution pattern on the network. Furthermore, as the network evolves in terms of the number of nodes and the number of users, the actual load distribution pattern may deviate largely from the expected one. Hence, this configuration provides a "worst-case" condition of network planning.
- The network may consist of several sub-networks, which are owned by multiple organizations, and consequently it becomes difficult to put a link capacity configuration for the whole network.

For brevity, we only consider the case in which resources are reserved in only one direction of the call (from calling party to called party). This is typical of real-time broadcast applications. The results of the more general case, in which resources are reserved in the two directions of the call has yielded similar results [2].

We start by introducing a relatively simple network model and simple traffic characteristics to allow us to explain the simulation results qualitatively. We then move to a more sophisticated network model in which we offer calls with more realistic traffic characteristics and delay requirements.

#### 4.1 Model 1: Merge-Split Network

In this model (see Fig. 1), calling group 1 and calling group 2 initiate calls to called group 1 and called group 2 respectively. Generated calls are distributed equally between calling group 1 and calling group 2.



**Fig. 1.** Topology of network model 1

The network topology suggests that non-even division policies can introduce an improvement in the blocking probability by putting more resource load on the four branching links ( $A \leftrightarrow D$ ,  $B \leftrightarrow D$ ,  $E \leftrightarrow F$ ,  $E \leftrightarrow G$ ) and thus increasing the number of calls that can be served by the bottleneck link  $D \leftrightarrow E$ .

In the absence of local stability, the number of calls that a switch can accept is limited by the minimum of the two bounds given by inequalities (3) and (4). However, only the scheduler stability bound limits the number of accepted calls at a switch when the reserved rates of accepted calls are lower than their average rates. We need to remove the switch stability bound when comparing the performance of different policies because they differ from each other in the reserved rate values. We do this by setting the average rate of incoming calls to zero whenever the reserved rate values for these calls are lower than the average rate used in simulations (32 Kbps). Those cases will be marked by (\*) in the simulation results.

For a better interpretation of the simulation outcome, the results show the average allocated rate at each switch, which is computed as the sum of rates reserved to all the accepted calls at the switch in a simulation run divided by their number.

The call load distribution for a total offered call load of ( $\rho$ ) is as follows:

- Call load on link  $D \leftrightarrow E = \rho$
- Call load on links  $A \leftrightarrow D, E \leftrightarrow G =$  call load from calling group 1  $= \rho/2$
- Call load on links  $B \leftrightarrow D, E \leftrightarrow F =$  call load from calling group 2  $= \rho/2$

The results show different values of the blocking probability corresponding to the source traffic burst size in ATM cells. The average allocated rate at each switch is in Kbps units. Simulation parameters are as follows:

Generated calls per simulation run = 100000	Access speed = 128 Kbps
Source average rate = 32 Kbps	Call load ( $\rho$ ) = 100 Erlangs
Required delay bound = 100msec	

**Configuration (A).** Here we take,  $C^1 = C^2 = C = 1.5$  Mbps

**Table 1.** Simulation results for configuration (A) of model 1

Burst Size	EVEN/CP		RCP			
	Blocking	Av. allocated rate	Blocking	Av. allocated rate		
				$D^1$	$A^1, E^1$	$B^1, E^2$
1*	0.06872	14.8	$<10^{-5}$	11.16	17.91	17.83
5	0.57815	34.55	0.55986	33.16	64.63	64.27
10	0.75385	59.23	0.75534	59.23	114.54	112.99
20	0.87078	108.58	0.87003	109.71	210.35	208.46

This configuration shows that RCP can provide an improvement over the EVEN policy. RCP allocates the rates in inverse proportion to the call load, i.e. the rate reserved on link  $D \leftrightarrow E$  is approximately half the rate allocated on the branching links. The results show that RCP gives a lower blocking probability than EVEN for smaller burst sizes, however, the RCP improvement over EVEN diminishes with the increase in burst size.

**Configuration (B).** Here we take,  $C^1 = C = 1$  Mbps,  $C^2 = 2C$

**Table 2.** Simulation results for configuration (B) of model 1

Burst Size	EVEN		CP			RCP		
	Blocking	Av. Allocated rate	Blocking	Av. allocated rate		Blocking	Av. allocated rate	
				$A^1, B^1, E^1, E^2$	$D^1$		$A^1, B^1, E^1, E^2$	$D^1$
1*	0.00348	14.8	0.23019	12.3	24.7	0.00205	14.4	16.8
5	0.45772	34.6	0.69169	32.2	64.3	0.56713	43.1	44.8
10	0.68772	59.4	0.83234	56.9	113.8	0.74368	73.3	75.7
20	0.82455	108.9	0.91053	106.4	212.8	0.85997	1131.2	136.3

The results show that the blocking probability of CP policy is higher than those of EVEN and RCP policies. This is because CP allocates a higher rate on the higher capacity and more loaded switch D1. This implies that CP defeats the main objective of proper network planning, which is to provide higher capacity to the more loaded links. We conclude that if a network is planned such that the capacity of each link is proportional to the expected call load offered to it, then CP policy should not be used.

**Configuration (C).** Here we take,  $C^1 = 2C$ ,  $C^2 = C = 1$  Mbps

**Table 3.** Simulation results for configuration (C) of model 1

Burst Size	EVEN		CP			RCP		
	Blocking	Av. allocated rate	Blocking	Av. allocated rate		Blocking	Av. allocated rate	
				$A^1, B^1, E^1, E^2$	$D^1$		$A^1, B^1, E^1, E^2$	$D^1$
1*	0.34602	14.8	0.06872	19.7	9.9	0.00032	26.5	8.0
5*	0.72378	34.6	0.67203	59.2	29.6	0.66369	110.6	28.9
10	0.83959	59.2	0.81855	108.6	54.3	0.82808	202.6	54.0
20	0.91053	108.6	0.91053	103.7	207.3	0.91265	362.2	104.7

The results show that CP/RCP performance is much better than EVEN policy. Yet, the performance gain decreases with the increase in burst size. We conclude that applying CP/RCP can prove useful in networks where actual call load pattern is drastically different from the anticipated one.



4.2 Model 2: Full Mesh 4-Nodes Network

In this section, a more sophisticated network model (see Fig. 2) is used. More realistic values of traffic characteristics are used in each simulation run. We also take the server stability limit into account, i.e. we don't set the source's average rate to zero when this rate is less than its reserved rate. In this model, calling group 1 and calling group 2 initiate calls to called group 1 and called group 2, respectively.

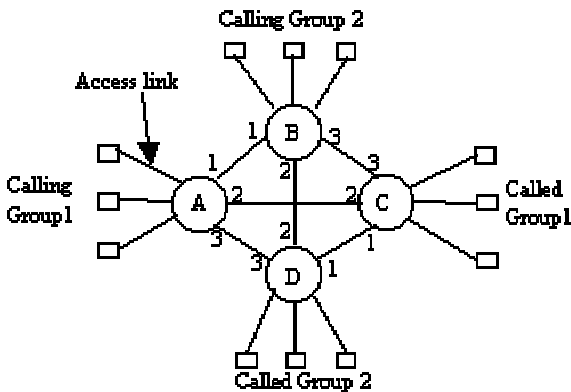


Fig. 2. Topology of network model 2

Generated calls are distributed equally between calling group 1 and calling group 2. Incoming calls from a calling group have five possible paths to the corresponding called group. The path of an incoming call is chosen at random from one of the five possible paths (i.e. random routing). Fig. 3 shows the call load pattern when using random routing.

The call load will be changed for each simulation run to keep the blocking probability in the order of  $10^{-3}$ - $10^{-5}$ . The offered traffic characteristic is selected as follows [3]:

- Average rate ( $r$ )=  $10^m$  Kbps,  $m$  is uniformly distributed on  $[0,3]$
- Burst size =  $y * r$  kbit,  $y$  is uniformly distributed on  $[0.5, 1.3]$
- Delay bound =  $50 * 10^s$  msec,  $s$  is uniformly distributed on  $[0, 1.52]$

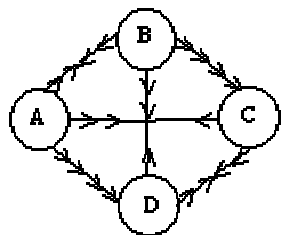


Fig. 3. Call load pattern with random routing

This range of generated traffic patterns include a typical MPEG video source with average rate = 518.4 kbps, and burst size = 576 kbit. It also includes a typical packetized voice source with average rate = 10 kbps, and burst size = 8 kbit. Link capacities are chosen from the standard Plesiochronous Digital Hierarchy (PDH) and Synchronous Digital Hierarchy (SDH) values. Access speed is chosen to be 6 Mbps, which is typical of ADSL (Asymmetric Digital Subscriber Line) units. The number of generated calls per simulation run is 100,000.

**Configuration (A).** Here we take the capacity of all links =  $C = 45$  Mbps (T3)

**Table 4.** Simulation results for configuration (A) of model 2

Call load	Blocking	
	EVEN/CP	RCP
400	0.00017	0.00238
420	0.00086	0.00462
440	0.00098	0.00913
460	0.00277	0.01284
480	0.00381	0.01923
500	0.00465	0.02435

**Configuration (B).** Here we assign capacities in proportion to the call load shown in Fig. 3. We will not be able to strictly apply the rule at all links because some links, such as link  $A \leftrightarrow C$  have different call loads in the reverse and forward directions and since we are assuming all the links to be symmetric (i.e. same capacity in the forward and backward paths), we will assign those links a capacity that is proportional to the higher call load. We thus have the following link capacity configuration:

$A \leftrightarrow B = A \leftrightarrow C = C \leftrightarrow D = D \leftrightarrow B = C$ , and  $C \leftrightarrow B = A \leftrightarrow D = 4C$  (instead of  $2C$  to match the speed factor used in PDH). ,  $C = 8.448$  Mbps (E2)

**Table 5.** Simulation results for configuration (B) of model 2

Call load	Blocking		
	EVEN	CP	RCP
60	0.00054	0.00701	0.00115
80	0.00152	0.0148	0.00458
100	0.00478	0.03004	0.01141
120	0.00908	0.03835	0.02023
140	0.01356	0.05069	0.03313
160	0.01871	0.06644	0.04655

**Configuration (C).** Here we assign capacities in inverse proportion to the call load shown in Fig. 3. We thus have the following link capacity configuration:  $A \leftrightarrow B = A \leftrightarrow C = C \leftrightarrow D = D \leftrightarrow B = 4C$ , and  $C \leftrightarrow B = A \leftrightarrow D = C$ ,  $C = 8.448$  Mbps (E2)

**Table 6.** Simulation results for configuration (C) of model 2

Call load	Blocking		
	EVEN	CP	RCP
20	0.00006	0.00006	0.00006
40	0.00107	0.00107	0.00157
60	0.00686	0.00588	0.00974
80	0.01501	0.01452	0.02149
100	0.02473	0.02379	0.03505
120	0.03596	0.03533	0.05169

**5 Enhancing Call Blocking Probability with Resource-based Routing**

This section presents the simulation results of model 2 when applying *Resource-based routing* in which the call path is selected as the least loaded path among the possible paths between the communicating parties. Except for configuration (A), where the number of generated calls per simulation run was taken to be 500,000 to allow accurate simulation of higher arrival rates associated with higher loads, all other parameters are the same as those of random routing simulations.

**Configuration (A).**

**Table 7.** Simulation results for configuration (A) of model 2, with resource-based routing

Call load	Blocking	
	EVEN/CP	RCP
780	0.000362	0.001382
800	0.0005	0.002508
820	0.000914	0.003128
840	0.00159	0.004544
860	0.00256	0.007326

Comparing tables 7,8, and 9 with tables 4,5, and 6, respectively shows the significant improvement introduced by the proposed resource-based routing. Note that comparable blocking probability values are achieved with much higher load values with resource-based routing.

**Configuration (B).****Table 8.** Simulation results for configuration (B) of model 2, with resource-based routing

Call load	Blocking		
	EVEN	CP	RCP
160	0.00003	0.03436	0.00006
180	0.00011	0.04173	0.00065
200	0.00029	0.05208	0.00162
220	0.00125	0.06039	0.00344
240	0.00251	0.06805	0.00688

**Configuration (C).****Table 9.** Simulation results for configuration (C) of model 2, with resource-based routing

Call load	Blocking		
	EVEN	CP	RCP
300	0.00002	0.00014	0.00001
320	0.00011	0.00057	0.00018
340	0.00067	0.00075	0.00075
360	0.00123	0.00271	0.00168
380	0.00295	0.00373	0.00331

**6 Conclusions**

We have studied resource allocation policies for ATM networks employing PGPS scheduling. We have addressed the problem of local mapping of end-to-end delay requirement into a local rate to be reserved at each switch along the path of an incoming call. Our findings can be summarized as follows:

1-For a network in which higher capacities are assigned to the links handling more call load, the CP policy is very inefficient because it allocates higher rates on those links. Furthermore, with larger burst sizes, CP cannot allocate smaller rates on lower capacity links and thus results in higher blocking.

2-In [7], it is argued that load imbalances can be viewed as resource imbalances, i.e., the node with a higher load may be thought of as a node with a load identical to other nodes but with smaller amounts of physical resources. Simulation results have revealed a flaw with the above statement. The imbalance in physical resources does not have the same effect as that of load imbalance because physical imbalance is a static effect, while load imbalance is a dynamic effect that depends on the network state. This explains why the amount of CP improvement (or deterioration) does not change with the offered call load while RCP does. This means that RCP can be better

than EVEN for some value of call load and worse for another one. Thus, measuring RCP performance must be done at the expected call load "operating point".

3-Non-even division of end-to-end QoS (for PGPS or any other scheduling discipline) results in unfairness among network paths as it increases the number of acceptable calls on a certain path at the expense of the other intersecting paths. This means that when a non-even division policy achieves a lower overall blocking probability than that for EVEN policy, it comes as a result of reducing the blocking probability of some paths, while increasing it for other paths but with less amount. This is in contrast to routing, which works to select the path with minimum loading and may be configured to aim at equalizing the call load among the different paths. Therefore, we suggest the use of EVEN policy for topologies with many intersecting paths (e.g. model 2), and the use of RCP for simpler topologies (e.g. model 1) with small number of intersecting paths. For the particular case of using PGPS scheduling, the improvement over EVEN policy diminishes with the increase in burst size and increases with the increase in the delay bound.

4-The use of resource-based routing greatly enhances the performance of all policies. This is achieved at the expense of having to employ a link-state protocol for distributing schedulers remaining capacity and an algorithm for determining the least loaded path. This may result in longer call-setup times.

## References

1. A. Ayad, K. Elsayed, M. El-Hadidi: Local Resource Allocation for Providing End-to-End Delay Guarantees in ATM Networks Using PGPS Scheduling. Proc. of Mascots'99 (1999)
2. A. Ayad: Resource Reservation for Deterministic end-to-end Delay Services in ATM Networks. M.Sc. Thesis. Faculty of Engineering. Cairo University (2000)
3. V. Firoiu, J. Kurose, D. Towley, Efficient Admission Control for EDF Schedulers. Proceedings of IEEE INFOCOM'97 (1997)
4. D. Ferrari, D. Verma: A Scheme for Real-Time Channel Establishment in Wide-Area Networks. IEEE JSAC, vol. 8, no. 4, pp. 368-379, April (1990)
5. P. Goyal, H.M. Vin: Generalized Guaranteed Rate Scheduling Algorithms: A Framework. TR-95-30, Department of Computer Sciences, University of Texas at Austin (1995)
6. S. Keshav: An Engineering Approach to Computer Networking: ATM Networks, the Internet, and the Telephone Network. (1995). Addison-Wesley
7. R. Nagarajan, J. Kurose, D. Towsley: Local Allocation of End-to-End Quality of Service in High-Speed Networks. Proceedings of IFIP Workshop on the Performance Analysis of ATM Systems, Martinique, Jan. (1993)
8. A.K. Parekh: A Generalized Processor Sharing Approach to Flow Control in Integrated Services Networks. Ph.D. dissertation, Dep. Elec Eng. Comput. Sci., M.I.T., Feb (1992)
9. S. J. Golestani: A Framing Strategy for Congestion Management. IEEE JSAC, vol. 9, no. 7, September (1991)

# Worst-Case Deterministic Delay Bounds for Arbitrary Weighted Generalized Processor Sharing Schedulers

Robert Szabó, Peter Barta, Felician Németh, and József Bíró

High Speed Networks Laboratory

Dept. of Telecommunications and Telematics, Technical University of Budapest

Stoczek u. 2, H-1111, Budapest, Hungary

{szabor,barta,nemethf,biro}@ttt-atm.ttt.bme.hu

**Abstract.** Generalized Processor Sharing (GPS) is realized as the most important ideal fluid scheduling discipline in guaranteed QoS networks. The well-known problem of GPS-based rate-proportional servers is the bandwidth delay coupling which leads to inefficient resource utilization. In this paper we introduce the concept of non rate-proportional (or arbitrary) weighting of sessions in GPS systems. Such an approach in a GPS node of network constrained by leaky buckets can handle bandwidth and delay parameters independently, thus allowing better utilization of network resources. Moreover, we show that even under the traditional bandwidth delay coupled system (rate-proportional weighting) it is possible to determine tighter delay bounds of sessions than that of presented in earlier papers. A numerically inexpensive algorithm, which works in any arbitrary weighted GPS system is also presented for computing delay bounds. Besides the analytical work numerical examples are also shown.

## 1 Introduction

The primary QoS requirements of applications in multi-service telecommunication networks such as end-to-end packet delay, loss and bandwidth must be provided on a per-connection basis for guaranteed performance services. A network meets these requirements primarily by appropriately scheduling its resources, which in turn assumes the use of proper traffic scheduling algorithms within individual network elements (switches, routers) [1,2,7].

Traffic scheduling algorithms usually operate on packet/cell level and assume there are different *sessions*, which traffic is to be scheduled for. The task of service disciplines at switching nodes is to control the order in which incoming packets are served in order to provide (end-to-end) *performance bounds*. Based on delay and fairness properties, one of the most significant scheduling algorithms is Generalized Processor Sharing (GPS) [5,6], which is a generalized version of Uniform Processor Sharing [4]. The packet-by-packet version of GPS (PGPS) is essentially the same as Weighted Fair Queuing (WFQ) [3], however they were independently developed.

A **GPS server** serving  $N$  sessions is characterized by  $N$  positive real numbers,  $\phi_1, \phi_2, \dots, \phi_N$ . The server operates at a *fixed rate*  $r$  and is work-conserving, that is, never idle whenever there are data to send. Let  $W_i(t_1, t_2)$  be the amount of session  $i$  traffic served in the interval  $[t_1, t_2]$ , then a GPS server is defined as one for which

$$\frac{W_i(t_1, t_2)}{W_j(t_1, t_2)} \geq \frac{\phi_i}{\phi_j}, \quad j = 1, 2, \dots, N \quad (1)$$

for any session  $i$  that is *continuously backlogged*<sup>1</sup> in the interval  $[t_1, t_2]$ . The immediate consequence of this definition is that every session has a minimum *guaranteed service rate*, that is  $g_i = \frac{\phi_i}{\sum_{j=1}^N \phi_j} r$  [5].

Under GPS scheduling each session has a separate queue served by the corresponding guaranteed rate in FIFO manner. The input traffic of every session can be shaped by a *leaky bucket* that is characterized by a token pool depth ( $\sigma$ ) and a token generation rate ( $\rho$ ). The amount of incoming traffic in the interval  $[t_1, t_2]$  from the active source  $i$ , assuming infinite capacity links, can be characterized by the function  $A_i(t_1, t_2)$ . If  $A_i(t) = A_i(0, t) = \sigma_i + \rho_i t$ , then by definition session  $i$  starts *greedy*, i.e., it starts with its maximal burst ( $\sigma_i$ ) at time zero and continues to transmit with its maximal rate ( $\rho_i$ ). If all sessions start greedy one gets a *greedy GPS system*. The main results of [5,6] is revisited here supporting our work:

*Suppose that  $C_j > r$  for every session  $j$ , where  $C_j$  is the internal link capacity between the session  $j$  leaky bucket and the session  $j$  queue, and  $r$  is the GPS server capacity. Then, for every session  $i$ , the maximum delay  $D_i^*$  and the maximum backlog  $Q_i^*$  are achieved (not necessarily at the same time) when every session is greedy starting at time zero, the beginning of a system busy period<sup>2</sup>. Further assuming that for each session  $i$   $g_i \geq \rho_i$ , then*

$$Q_i^* \leq \sigma_i \quad \text{and} \quad D_i^* \leq \frac{\sigma_i}{g_i}. \quad (2)$$

The significance of the former result is that for worst-case behavior one ‘only’ has to analyze a greedy system, which makes the analysis more tractable compared to any arbitrary arrival patterns imposed to the system. *Therefore, hereafter in the paper without losing generality we only consider greedy GPS systems, where all sessions start greedy at time zero, the beginning of a system busy period.* Since the arrival function in the case of infinite capacity links upper bounds the finite capacity link case, throughout our analytical approach we consider only the former scenario. The transformation of the results obtained under the infinite capacity links to the finite one can be done similarly as shown in [5].

A large subclass of GPS-based schedulers is the rate-proportional servers (RPS), in which weights are set according to session bandwidth demands. It introduces coupling between bandwidth and delay, i.e., the worst-case delay bound is inversely proportional to the long term allocated bandwidth ( $\rho$ ). The drawback with this weighting is that if one wants to decrease the delay bound of a

<sup>1</sup> a session is said to be backlogged if there are some data in its queue

<sup>2</sup> an interval the server is continuously working

session then the corresponding bandwidth should be increased in order to maintain rate-proportional bandwidth allocation. Clearly, this results in inflexible resource allocation and may lead to waste of network resources.

As an example, assume that one would like to half the worst-case delay for session  $i$ . Since the worst-case delay is  $\sigma_i/g_i$  where  $g_i = \phi_i/\sum_j \phi_j$ ,  $\phi_i$  should be doubled, which also results in double guaranteed rate of service for that session. In this way, due to the rate-proportional weighting of sessions, the long term sustainable throughput  $\rho_i$  should also be multiplied by two. This concludes that the more delay sensitive a session is the more bandwidth it requires. Nevertheless, there can be sessions like voice over IP (VoIP), which are delay sensitive but do not require high bandwidth compared to other sessions. In this case, bandwidth (server capacity) might be wasted for providing acceptable upper bound for delay.

The problem is two-fold. On one hand, the delay bounds for sessions are quite loose, although they are also very simple (see (2)). On the other hand, these bounds are inversely proportional to the minimum guaranteed service rate and also to the long term sustainable rate due to the rate-proportional weighting of sessions.

In our approach we propose solutions for both of the above problems. As opposed to the rate-proportional weighting we consider a case when the weight  $\phi_i$  is set independently of the parameter  $\rho_i$ . This is referred to as *non rate-proportional*<sup>3</sup> weighting of sessions. After characterizing the behaviour of such a system we provide an algorithmic approach to compute *tighter* upper bounds for delay than those were derived in [5,6]. This algorithm universally applies to any arbitrary weight assignment, but due to the fact that only the rate-proportional weighting was studied in [5,6], the tightness of our delay bound cannot be compared in case of arbitrary weighting.

The rest of the paper is organized as follows. In the next section we introduce a novel approach to characterize the service rates of an arbitrary weighted GPS server. Next we present an algorithm to calculate useful (tighter) bounds for worst-case delay based on the dynamics of non rate-proportional GPS server. Finally, a numerical example illustrate our theoretical work and the final section draws conclusions.

## 2 Service Rates of a GPS Server

Now, let us turn on a more formal description of the system behaviour. For simplicity let  $r = 1$ , in other words we consider all the quantities normalized to the server rate  $r$ . Let  $\mathcal{L} = \{L(i) \mid i = 1, \dots, N\}$  be an ordered set of indices in which  $L(i)$  means that the session  $L(i)$  backlog is cleared as  $i^{\text{th}}$  in order. Further, let us denote the time instant by  $t_i$  when the backlog of session  $L(i)$  becomes zero. By definition, let  $t_0 = 0$  the start of the system busy period with all greedy sessions. During the interval  $[0, t_1)$  every session is served by its

<sup>3</sup> non rate-proportional and arbitrary is used interchangeably through this article, though arbitrary weighting contains rate-proportional weighting as a special case



minimum guaranteed service rate, i.e.,  $r_1^j = g_j$ . Between  $t_1$  and  $t_2$  session  $L(1)$  has no backlog, so it is served by  $\rho_{L(1)}$ . The service rates of the still backlogged sessions during that period are

$$r_2^j = g_j + \frac{\phi_j}{\sum_{i=1}^N \phi_i - \phi_{L(1)}} (g_{L(1)} - \rho_{L(1)}), \quad j \in \{1, \dots, N\} \setminus \{L(1)\} . \quad (3)$$

The second part on the right hand side comes from the fact that after clearing the backlog of session  $L(1)$  the remaining bandwidth  $(g_{L(1)} - \rho_{L(1)})$  is distributed among the other sessions in proportion to their weights.

**Theorem 1.** *During the time interval  $[t_{k-1}, t_k]$ ,  $k = 1, \dots, N$  within the system busy period the backlogged session  $j$  is served at a rate*

$$r_k^j = \frac{(1 - \sum_{l=1}^{k-1} \rho_{L(l)}) \phi_j}{\sum_{i=1}^N \phi_i - \sum_{l=1}^{k-1} \phi_{L(l)}} , \quad j \in \{1, \dots, N\} \setminus \left\{ \bigcup_{m=1}^{k-1} L(m) \right\} . \quad (4)$$

*Proof.* The theorem is proved by induction. First, we check that the statement is valid for  $r_m^j$  when  $m = 2$ . From equation (3)

$$r_2^j = \frac{\phi_j}{\sum_{i=1}^N \phi_i} + \frac{\phi_j}{\sum_{i=1}^N \phi_i - \phi_{L(1)}} \frac{\phi_{L(1)}}{\sum_{i=1}^N \phi_i} - \frac{\phi_j}{\sum_{i=1}^N \phi_i - \phi_{L(1)}} \rho_{L(1)} . \quad (5)$$

Since

$$\frac{\phi_j}{\sum_{i=1}^N \phi_i} + \frac{\phi_j}{\sum_{i=1}^N \phi_i - \phi_{L(1)}} \frac{\phi_{L(1)}}{\sum_{i=1}^N \phi_i} = \frac{\phi_j}{\sum_{i=1}^N \phi_i - \phi_{L(1)}} \quad (6)$$

the following equality holds

$$r_2^j = \frac{(1 - \rho_{L(1)}) \phi_j}{\sum_{i=1}^N \phi_i - \phi_{L(1)}} . \quad (7)$$

Now, let us assume that the statement in the theorem is valid for  $m = k - 1$ , which means that

$$r_{k-1}^j = \frac{(1 - \sum_{l=1}^{k-2} \rho_{L(l)}) \phi_j}{\sum_{i=1}^N \phi_i - \sum_{l=1}^{k-2} \phi_{L(l)}} , \quad j \in \{1, \dots, N\} \setminus \left\{ \bigcup_{v=1}^{k-2} L(v) \right\} . \quad (8)$$

The question is how this rate changes when the session  $L(k - 1)$  backlog is cleared at time instant  $t_{k-1}$ . The session  $L(k - 1)$  service rates are  $r_{k-1}^{L(k-1)}$  and  $\rho_{L(k-1)}$  respectively before emptying its backlog and after emptying its backlog. In this way the remaining capacity  $r_{k-1}^{L(k-1)} - \rho_{L(k-1)}$  is distributed among sessions still backlogged in the system in proportion to their weights. So one can write

$$r_k^j = r_{k-1}^j + \frac{\phi_j}{\sum_{i=1}^N \phi_i - \sum_{l=1}^{k-1} \phi_{L(l)}} (r_{k-1}^{L(k-1)} - \rho_{L(k-1)}),$$

where  $j \in \{1, \dots, N\} \setminus \left\{ \bigcup_{v=1}^{k-1} L(v) \right\} . \quad (9)$

Substituting  $r_{k-1}^j$  from equation (8) it is obtained that

$$r_k^j = \frac{(1 - \sum_{l=1}^{k-2} \rho_{L(l)})\phi_j}{\sum_{i=1}^N \phi_i - \sum_{l=1}^{k-2} \phi_{L(l)}} + \frac{\phi_j}{\sum_{i=1}^N \phi_i - \sum_{l=1}^{k-1} \phi_{L(l)}} \left( \frac{(1 - \sum_{l=1}^{k-2} \rho_{L(l)})\phi_{L(k-1)}}{\sum_{i=1}^N \phi_i - \sum_{l=1}^{k-2} \phi_{L(l)}} - \rho_{L(k-1)} \right). \quad (10)$$

After some manipulations one gets

$$r_k^j = \frac{(1 - \sum_{l=1}^{k-2} \rho_{L(l)})\phi_j}{\sum_{i=1}^N \phi_i - \sum_{l=1}^{k-1} \phi_{L(l)}} - \rho_{L(k-1)} \frac{\phi_j}{\sum_{i=1}^N \phi_i - \sum_{l=1}^{k-1} \phi_{L(l)}}, \quad (11)$$

which yields the statement of Theorem 1 after simplification.  $\square$

**Corollary 1.**  $r_k^j$  is an increasing function of  $k$  for every backlogged session.

*Proof.* Using the formula of (9) we only have to prove that the increment in the iterative formula is greater than or equal to zero. In this case the first factor of the multiplications is strictly greater than zero. Since  $L(k-1)$  identifies the session that last emptied its backlog, its backlog function can be expressed in a form

$$Q_{L(k-1)}(t) = Q_{L(k-1)}(\tau) + \rho_{L(k-1)}(t - \tau) - r_{k-1}^{L(k-1)}(t - \tau) \quad (12)$$

where:  $Q_{L(k-1)}(\vartheta)$  is the session  $L(k-1)$  backlog (in units of traffic) at time  $\vartheta$  and  $\tau$  is the time when session  $L(k-2)$  emptied its backlog.

Since at time  $\tau$  session  $L(k-2)$  emptied its backlog and session  $L(k-1)$  is still backlogged, so  $Q_{L(k-1)}(\tau) > 0$ . On the other hand, the next session to empty its backlog is  $L(k-1)$ , so for some  $t > \tau$ ,  $Q_{L(k-1)}(t)$  will be zero, thus yielding for the inequality  $r_{k-1}^{L(k-1)} > \rho_{L(k-1)}$ , which proves our corollary.  $\square$

In order to introduce a theorem defining the maximum backlog of sessions we first declare two lemmas each stating an important attribute of a GPS system.

**Lemma 1.** For a system busy period in a stable system ( $\sum_i \rho_i < r$ ) where all sessions start greedy, at time zero (beginning of a system busy period) there is at least one session that starts its service with higher rate than its arrival rate.

*Proof.* This Lemma is proved by contradiction. For a stable system  $\sum_{i=1}^N \rho_i < r$  [5] and the initial service rate, i.e., the guaranteed rate for each session is defined as  $g_i = \frac{\phi_i}{\sum_{i=1}^N \phi_i} r$ . Now assume there exists no session whose guaranteed service rate is higher than its arrival rate, i.e.,  $g_i \leq \rho_i$ ,  $\forall i \in \{1, \dots, N\}$ .

$$r > \sum_{i=1}^N \rho_i \geq \sum_{i=1}^N g_i = r \quad (13)$$

which is a contradiction, thus ensuring the existence of at least one session with strictly higher service rate than its arrival rate that proves our Lemma.  $\square$

**Lemma 2.** *In a system busy period where all sessions start greedy, at any time  $t$  there is at least one backlogged session that is served with higher rate than its arrival rate.*

*Proof.* This Lemma is proved by contradiction. At any time  $t$  during the system busy period sessions can be divided into two disjoint sets; one corresponding to those already emptied their backlogs  $\mathcal{F}(t)$  and the other still having backlogs at time  $t$  further denoted by  $\mathcal{B}(t)$ . At any time  $t$   $\mathcal{F}(t) \cap \mathcal{B}(t) = \emptyset$  and  $\mathcal{F}(t) \cup \mathcal{B}(t) = \{1, \dots, N\}$ . We still have the stable system criterion, i.e.,  $\sum_{i=1}^N \rho_i < r$ , however this can be reorganized according to the two sets:  $\sum_{i \in \mathcal{F}(t)} \rho_i + \sum_{i \in \mathcal{B}(t)} \rho_i < r$ .

Further denote session  $i$  service rate at time  $t$  by  $r_i(t)$  in a way that for  $\forall k \in \{1, \dots, N\}$

$$r_i(t) = r_k^i, \text{ if } t \in [t_{k-1}, t_k) . \quad (14)$$

Now assume that there exists a  $t$  that  $r_i(t) \leq \rho_i \forall i \in \mathcal{B}(t)$ , while  $r_i(t) = \rho_i \forall i \in \mathcal{F}(t)$ .

$$r - \sum_{i \in \mathcal{F}(t)} \rho_i > \sum_{i \in \mathcal{B}(t)} \rho_i \geq \sum_{i \in \mathcal{B}(t)} r_i(t) . \quad (15)$$

From Theorem 1 one can write

$$r_i(t) = \frac{\phi_i}{\sum_{j \in \{\mathcal{B}(t) \cup \mathcal{F}(t)\}} \phi_j - \sum_{j \in \mathcal{F}(t)} \phi_j} (r - \sum_{j \in \mathcal{F}(t)} \rho_j) \quad (16)$$

that is summed for all  $i \in \mathcal{B}(t)$  will yield for

$$\begin{aligned} \sum_{i \in \mathcal{B}(t)} r_i(t) &= \frac{\sum_{i \in \mathcal{B}(t)} \phi_i}{\sum_{i \in \mathcal{B}(t)} \phi_i + \sum_{j \in \mathcal{F}(t)} \phi_j - \sum_{j \in \mathcal{F}(t)} \phi_j} (r - \sum_{j \in \mathcal{F}(t)} \rho_j) \\ &= (r - \sum_{j \in \mathcal{F}(t)} \rho_j) . \end{aligned} \quad (17)$$

From (17) the contradiction of (15) immediately follows.  $\square$

**Theorem 2.** *Each session  $i$  experiences its maximal backlog either at time zero or at a time when its rate increases (changing).*

*Proof.* From Corollary 1 it follows that there are at most two phases for each session: the first corresponds to the accumulation phase, where the backlog of session  $j$  increases and the second that corresponds to backlog emptying phase.

Those sessions starting their services with higher service rates than their arrival rates will only take part in the backlog emptying phase. Since after their initial burst is cleared their backlogs will continuously decrease and their maximal backlog is limited to their initial burst size.

The other set of sessions, which start with smaller service rates than their arrival rates, will step by step increase their rates by each time a session empties its backlog (result of Corollary 1).

From Lemma 2 it further follows that there's always at least one session clearing its backlog, thus the set of backlogged sessions will decrease ensuring the transition of accumulating sessions to backlog clearing ones. These sessions will experience their maximum backlog at these transition times.  $\square$

### 3 Computing Backlog Clearing Times

In the previous section we have seen that the time instances when the backlogs are cleared and their orders play a key role in the dynamics of the traffic service. Here we provide an algorithmic approach how to compute the backlog clearing times  $t_k$ ,  $k = 1, \dots, N$  and based on these quantities a tighter upper bound for the maximum delay of every session.

As the main result of [5] the maximum backlog and delay of sessions occur (not necessarily at the same time) when all sessions are greedy, i.e., they start to send data at the same time and with maximum possible rate.

In the first step, we should consider the following  $N$  equations, which describe the time-evolution of incoming and outgoing (being served) traffic of sessions.

$$\sigma_i + \rho_i(t - t_0) = r_1^i(t - t_0) \ , \ i = 1, \dots, N \ . \quad (18)$$

On the left-hand side the incoming traffic of session  $i$  is expressed until time  $t$  and on the right-hand side the amount of served traffic is represented until time  $t$  provided the service rate is  $r_1^i$ . Assuming the former service rate and the definition of  $t_0 = 0$ , the time needed for session  $i$  to empty its backlog is expressed

$$t_{i,1} = \frac{\sigma_i}{r_1^i - \rho_i} \ . \quad (19)$$

It is clear that among  $t_{i,1}$ 's there can be negative values in case of those sessions whose backlog is temporarily increasing. Apparently, that session clears its backlog first whose  $t_{i,1}$  takes the smallest positive value. More formally

$$t_1 = \min\{t_{i,1} \mid t_{i,1} > 0; \ i \in \{1, \dots, N\}\} \quad (20)$$

and  $L(1)$  is the index of session  $i$  that satisfies the above minimum.

After session  $L(1)$  empties its backlog at time  $t_1$ , there remain  $N - 1$  backlogged sessions in the server. The service rate of session  $i$  backlogged is changed (increased) to  $r_2^i$ . In the next (second) step for determining  $t_2$  consider the following  $N - 1$  equations:

$$\sigma_i + \rho_i(t - t_0) = r_1^i(t_1 - t_0) + r_2^i(t - t_1), \ i \in \{1, \dots, N\} \setminus L(1) \ . \quad (21)$$

After decomposing  $\rho_i(t - t_0)$  and solving the equation for  $t$  one gets

$$t_{i,2} = \frac{\sigma_i - r_1^i t_1 + r_2^i t_1}{r_2^i - \rho_i} \quad (22)$$

as the candidate finishing times of still backlogged sessions. Taking the minimum of positive backlog times through  $i \in \{1, \dots, N\} \setminus L(1)$  will yield for the next finishing time of the system

$$t_2 = \min\{t_{i,2} \mid t_{i,2} > 0; \ i \in \{1, \dots, N\} \setminus L(1)\}, \quad (23)$$

where  $L(2)$  corresponds to the session index  $i$  that satisfies the above minimum.

In general, the  $k^{\text{th}}$  step can be expressed by the equation

$$\sigma_i + \sum_{j=1}^{k-1} \rho_i(t_j - t_{j-1}) + \rho_i(t - t_{k-1}) = \sum_{j=1}^{k-1} r_j^i(t_j - t_{j-1}) + r_k^i(t - t_{k-1}) \ . \quad (24)$$

Solving the above equation for  $t$  yields the  $k^{\text{th}}$  step candidate finishing times of still backlogged sessions, i.e.,

$$t_{i,k} = \frac{\sigma_i + \sum_{j=1}^{k-1} (\rho_i - r_j^i)(t_j - t_{j-1}) + (r_k^i - \rho_i)t_{k-1}}{r_k^i - \rho_i} \ . \quad (25)$$

Since  $t_0 = 0$  by definition,  $\rho_i$ 's can be eliminated from the numerator yielding for the following result

$$t_{i,k} = \frac{\sigma_i + \sum_{j=1}^{k-1} r_j^i(t_{j-1} - t_j) + r_k^i t_{k-1}}{r_k^i - \rho_i} \ . \quad (26)$$

The  $k^{\text{th}}$  step finishing time is calculated by taking the minimum of (26) over  $i \in \{1, \dots, N\} \setminus \bigcup_{j=1}^{k-1} L(j)$ , i.e.,

$$t_k = \min\{t_{i,k} \mid t_{i,k} > 0; \ i \in \{1, \dots, N\} \setminus \bigcup_{j=1}^{k-1} L(j)\} \ . \quad (27)$$

However, to reduce the computation complexity of subsequent  $t_{i,k}$ 's, we found the following recursion:

$$t_{i,k} = \frac{S_{i,k}}{r_k^i - \rho_i}, \quad (28)$$

where  $S_{i,k}$  is the numerator of (26) and

$$S_{i,k+1} = S_{i,k} - r_k^i t_k + r_{k+1}^i t_k = S_{i,k} + (r_{k+1}^i - r_k^i) t_k \ . \quad (29)$$

The calculation of backlog clearing times is quite simple and can be highly automated using our proposed recursive formula. One only has to maintain the rolling sum of  $S_{i,k}$ 's, then calculating  $r_k^i$  in each step using (9) and further taking the minimum of (28) to get the next step finishing time.

## 4 Calculating Tighter Upper Bounds for Delay

The delay bounds of GPS systems - or its packetized versions - have been widely analyzed under leaky-bucket constrained input traffics. However, the simple delay bound (see (2))

$$\hat{D}_i = \frac{\sigma_i}{g_i}, \quad (30)$$

where  $g_i$  is the guaranteed service rate of session  $i$  holds only when  $g_i \geq \rho_i$  for session  $i$  [6,7]. These sessions are called locally stable sessions, i.e., they do not have backlog accumulating phase [6].

This very constraint however does not allow to set weights ( $\phi$ ) without restrictions, thus limiting the usable parameter space.

To overcome this limitation, we introduce a theorem to calculate the delay of sessions for any arbitrary weighting. Further, we state that the delay bound we calculate is at least as good as that given by (30), provided locally stable weight assignment.

### Theorem 3 (Delay Bounds).

(i) If  $r_i(t)$  as defined in (14) satisfies

$$r_i(\tau_i) \geq \rho_i \quad (31)$$

then

$$D_i = \tau_i, \quad (32)$$

where  $\tau_i$  is defined:

$$W_i(0, \tau_i) = \int_0^{\tau_i} r_i(t) dt = \sigma_i. \quad (33)$$

(ii) If (31) does not hold for session  $i$  then it starts with an accumulating phase and denote the order it clears its backlog by  $I : i = L(I)$ , further there exists a  $J \in \{1, \dots, I-1\}$  such that

$$\{\forall t < t_J : r_i(t) < \rho_i\} \wedge \{\forall t \geq t_J : r_i(t) \geq \rho_i\} \quad (34)$$

holds, then

$$D_i = t_J - \frac{W_i(0, t_J) - \sigma_i}{\rho_i}, \quad (35)$$

where  $W_i(0, t)$  is the amount of session  $i$  traffic served up till time  $t$ .

*Proof.* We want to calculate the maximal delay of session  $i$ , i.e.,

$$D_i = \max_t \{t - A_i^{-1}(W_i(t))\}, \quad (36)$$

where  $A_i^{-1}(W)$  is the inverse of the arrival function. As we consider worst-case maximal delay, we assume a greedy system, where the arrival function and its inverse are

$$A_i(t) = \sigma_i + \rho_i t \quad (37)$$

$$A_i^{-1} = t(A_i) = \max \left\{ \frac{A_i - \sigma_i}{\rho_i}, 0 \right\} . \quad (38)$$

Now the delay of session  $i$  can be written as

$$D_i(t) = t - \max \left\{ \frac{\int_0^t r_i(\tau) d\tau - \sigma_i}{\rho_i}, 0 \right\}, \forall t \geq 0 . \quad (39)$$

Since (39) is an increasing function for  $t \in [0, \tau_i]$  where  $\tau_i$  is defined by (33), the smallest  $t$  when  $D_i(t)$  achieves its maximum is  $\tau_i$ , so (39) can be rewritten as

$$D_i(t) = t - \frac{\int_0^t r_i(\tau) d\tau - \sigma_i}{\rho_i}, \forall t \geq \tau_i . \quad (40)$$

The delay function ( $D_i(t)$ ) is continuous and linear between breaking points determined by the backlog clearing times. Hence, it's maximum is achieved at a breaking point where its left and right hand side derivate change sign. Thus, taking the left and right hand side derivation of (40) with respect to  $t$ , one gets the maximum where the derivate turns from positive to negative. The left hand side derivate is

$$D_i'^-(t) = 1 - \frac{r_i^-(t)}{\rho_i} \quad (41)$$

and the right hand side one is expressed similarly, thus the condition of sign changing can be rewritten as

$$r_i^-(T_i^*) < \rho_i, \quad r_i^+(T_i^*) > \rho_i . \quad (42)$$

Since  $r_i(t)$  is an increasing function in  $t$  (see Corollary 1), there are two cases:

(a) If  $r_i(\tau_i) \geq \rho_i$ , then (40) is a decreasing function, so the maximal delay is achieved at time  $\tau_i$ , which proves (i).

(b) If  $r_i(\tau_i) < \rho_i$ , then  $T_i^*$  satisfying (42) is greater than  $\tau_i$ . However as  $r_i(t)$  is a step function (see Theorem 1) we seek for a  $t_J$  finishing time that satisfies (34). Further this  $t_J$  will also maximize the delay, since the step function of  $r_i(t)$  will cross  $\rho_i$  at  $t_J$ , which proves (ii).  $\square$

Now we have to prove that our delay bound is at least as good as (30).

**Theorem 4.** *We state that*

$$D_i \leq \hat{D}_i, \quad (43)$$

where  $D_i$  is defined by Theorem 3 and  $\hat{D}_i$  is defined by (30).

*Proof.* Since  $\hat{D}_i$  holds only if  $g_i \geq \rho_i$ , from Theorem 3 we only have to consider case (i). From Corollary 1 it follows that

$$r_i(t) \geq g_i, \quad \forall t \in [0, \tau_i], \quad (44)$$

where  $\tau_i$  is defined by (33), thus replacing  $r_i(t)$  by  $g_i$  in (33) we get

$$\int_0^{\hat{\tau}_i} g_i dt = g_i \hat{\tau}_i = \sigma_i . \quad (45)$$

Now using (44), (45) and Theorem 3, one can write  $D_i = \tau_i \leq \hat{\tau}_i = \frac{\sigma_i}{g_i} = \hat{D}_i$ , that proves our Theorem.  $\square$

**Table 1.** Parameters and delays of two sessions

Session parameters	$\sigma$	$\rho$	$\phi^1$	$\phi^2$	$\phi^3$	$\phi^4$	$\phi^5$	$\phi^6$
Session 1 (dashed line)	1	.2	2	6	1	12	1	100
Session 2 (cont. line)	3	.6	6	2	12	1	100	1
<b>Delays</b>								
$D_1$			4	4/3	10	13/12	10	101/100
$D_2$			4	5	13/4	5	303/100	5

## 5 Numerical Example

In this section we show a numerical example on our approach to calculate delay and backlog bounds for a GPS node.

Consider a GPS node serving two greedy sessions. The parameters describing the sessions and the applied weights can be seen in Table 1. Note that the set  $\phi^1$  corresponds to the traditional case where weights are assigned proportional to the long term sustained rates.

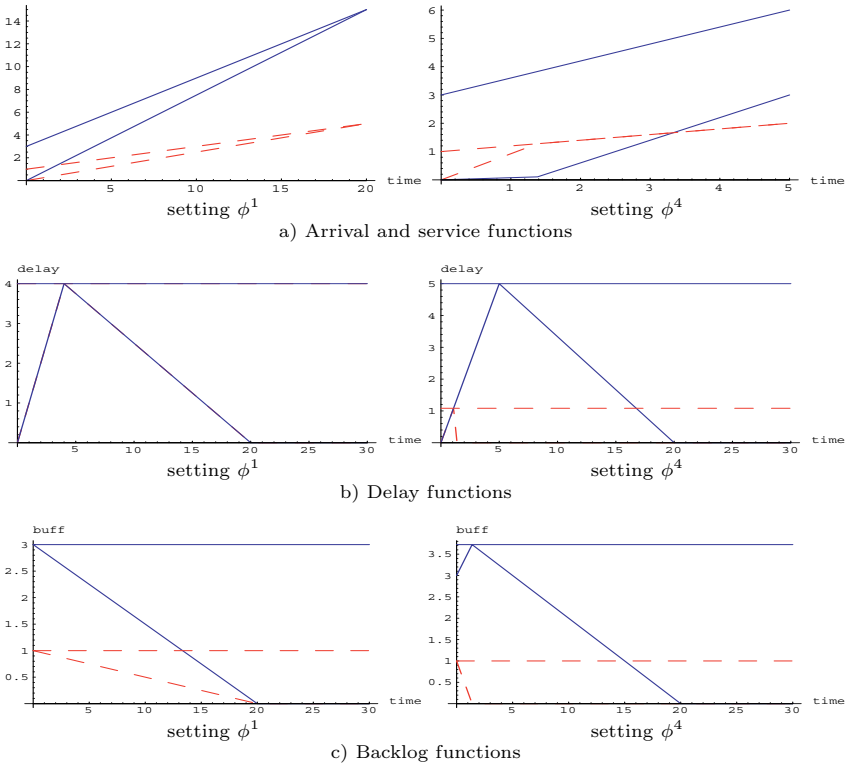
Table 1 also shows the calculated delay values using our formula. This gives the base of our comparison. It strikes out immediately that the achievable delays are limited to certain intervals, i.e.,  $D_1 \in (1, 10]$  and  $D_2 \in (3, 5]$ . Despite the correlation of the two delays, one can favor any of the sessions to come close to its theoretical minimal delay.

To better understand the the behavior of the GPS node, we present results on the arrival and service functions in Fig. 1.a. Note that same sessions are drawn with same line styles and service functions start from zero while arrival functions start with bursts<sup>4</sup>. The figure nicely shows how service rate is increasing at backlog emptying time. The fact that one of the sessions sooner or later empties its backlog and further consumes resources (bandwidth) only as much as its sustained rate ( $\rho$ ) allows the other session to clear its backlog faster, thus reducing its delay. The different settings of weights determine whose backlog is emptied first.

Since our primary goal was to introduce tighter delay bound, let us investigate the delay behavior of the sessions. Fig. 1.b shows the maximal and the experienced delays in the function of time. Again, the same line style corresponds to one session's maximal delay and delay function in time. It is eye-catching that the delay curves for  $\phi^1$  coincides. However there may be situations when lower rate sessions have real time delay requirements. Now one has to de-couple bandwidth demands from the actual weights and set  $\phi$ 's according to the delay needs. For example, in setting  $\phi^4$  the small bandwidth session is favored against the high bandwidth one. In this case one may also consider that the increase in worst-case delay for the high bandwidth traffic (from 4 to 5) may worth the significant decrease of low rate traffic's delay (from 4 to 13/12). Further, one

<sup>4</sup> the characteristics of session 1 and session 2 are denoted by dashed and continuous lines respectively





**Fig. 1.** Session characteristics of an arbitrary weighted GPS node

may carry out a system that is fair in a sense, that smoother traffic gets lower worst-case delay performance than burst ones with indifference of their rates.

It is usually said that there are no gains without losses. The price we have to pay for playing with the weights and setting them arbitrarily is the increase of necessary buffer capacity. Since in the bandwidth delay coupled system the guaranteed service rates are greater than the corresponding arrival rates ( $g_i \geq \rho_i$ ) session backlogs are determined by the burst sizes (see Fig. 1.c setting  $\phi^1$ ). On the other hand when setting weights de-coupled one of the sessions has to suffer higher delay thus requiring bigger buffers (see Fig. 1.c setting  $\phi^4$ ).

## 6 Concluding Remarks

In this paper we have analyzed the GPS service discipline with arbitrary weighting of sessions. Although the analysis has been performed for an ideal (fluid) service model the results can be extended to packet-by-packet schedulers in a straightforward manner. We proposed a novel approach to calculate tighter delay bounds under the traditional rate-proportional GPS systems. These bounds are generally tighter than those obtained by Parekh and Gallager in [5], be-

cause the actual service rate of sessions is taken into account instead of using the guaranteed rate only. The developed algorithm also works in any arbitrary weighted GPS schedulers. A numerical example was presented to detail the behavior of such schedulers. In addition, we have demonstrated that by using arbitrary weighting of sessions the bandwidth delay coupling can be relaxed and worst-case delay bounds can arbitrarily be set between the theoretical bounds.

Besides the tighter delay bound computation under rate-proportional weighting, our work placed emphasis on the assurance of delay bound requirements in such a way that the server capacity is not wasted. This flexibility is achieved by the introduction of non rate-proportional weighting, which also means that the guaranteed fairness may not be valid on all time scale. However, as long as the server is stable, the long term sustainable throughput ( $\rho$ ) is guaranteed for each session, that is the long term guaranteed fairness is ensured.

The significance of bandwidth delay decoupling and tight performance bounds in existing guaranteed QoS networks is evident. Furthermore, in future Internet services efficient schedulers with predictable performance will also play a key role. Our results can be an important contributions to service models proposed for QoS Internet, too.

## Acknowledgement

This work has been performed in the joint-cooperation between Ericsson - Applied Research Switch Lab and High-Speed Networks Laboratory, Department of Telecommunications and Telematics, Technical University of Budapest. The continuous support of Ericsson is highly appreciated.

## References

1. J.C.R. Bennett and H. Zhang. Hierarchical packet fair queueing algorithms. In *Proceedings ACM SIGCOMM'96*, pages 143–156, August 1996.
2. D. Clark, S. Shenker, and L. Zhang. Supporting real-time applications in an integrated services packet network: Architecture and mechanism. In *Proceedings of ACM SIGCOMM'92*, pages 14–26, Baltimore, Maryland, August 1992.
3. Alan Demers, Srinivasan Keshav, and Scott Shenker. Analysis and simulation of a fair queueing algorithm. In *SIGCOMM Symposium on Communications Architectures and Protocols*, pages 1–12, Austin, Texas, September 1989. ACM. also in *Computer Communications Review*, 19 (4), Sept. 1989.
4. L. Kleinrock. *Queueing Systems, Volume 2: Computer Applications*. Wiley, 1976.
5. A.K. Parekh and R. G. Gallager. A generalized processor sharing approach to flow control in integrated services networks: The single-node case. *IEEE/ACM Transactions on Networking*, 1(3):344–357, June 1993.
6. A.K. Parekh and R. G. Gallager. A generalized processor sharing approach to flow control in integrated services networks: The multiple node case. *IEEE/ACM Transactions on Networking*, 2(2):137–150, April 1994.
7. H. Zhang. Service disciplines for guaranteed performance service in packet-switching networks. In *Proceedings of the IEEE*, 83(10), October 1995.

# Deterministic End-to-End Delay Bounds in an Accumulation Network

Guillaume Urvoy<sup>1</sup>, Gérard Hébuterne<sup>1</sup>, and Yves Dallery<sup>2</sup>

<sup>1</sup> Institut National des Télécommunications,  
Département RST, 9, rue Charles, Fourier 91011 Evry, France

<sup>2</sup> Laboratoire d'Informatique de Paris 6,  
8, rue Capitaine Scott, 75015 Paris, France

**Abstract.** In this paper, we focus on the determination of end-to-end delay bounds for FIFO accumulation networks with leaky bucket constrained sources, including IP and ATM network cases. We determine an upper bound for the overall end-to-end delay and prove its accuracy to approximate the exact maximum end-to-end delay for accumulation networks of any size. This is achieved through an original trajectory analysis approach. Numerical studies further illustrate this point. This extends our previous results for the two-server and proves that this bound may be used as an accurate criterion for a CAC algorithm providing a deterministic QoS guaranteed service.

## 1 Introduction

The design and operation of multi-service networks providing Quality of Service (QoS) is a challenge in which Call Admission Control (CAC) algorithms are the key issue for non-adaptive traffic sources. Invoking the CAC procedure consists mainly in two basic procedures : determining a bound (on loss rate or delay for instance) on the path of the new source and checking the non-violation of the QoS of already established connections.

In this paper, we focus on the determination of deterministic end-to-end delay bounds in a FIFO accumulation network. Note that by doing so, we do not address the entire CAC problem, as the latter may not be reduced to the determination of some bounds. Most of the studies avoid this problem because they model the network through a single server (refer to the algorithms presented in [9,8]) whereas CAC is an end-to-end issue. An accumulation or concentration network is a tree network where sources may enter at any node but exit at the root node only. Though not the case in general, this topology is not only of theoretical interest. For instance, a Multipoint-to-Point Tree (MPT) is considered in [1] for the routing problem of IP over ATM networks.

We assume a fluid flow model and leaky bucket constrained sources. This model perfectly fits ATM networks (for VBR sources) due to the small cell size compared to the servers' rates. The results we obtain in the present work may also be applied to IP networks (for sources declared via a TSpec [4]) with some more

caution due to the variable size of the packets (see for instance [7] for the translation between a fluid and a packet model).

The remainder of this paper is organized as follows. In Sect. 2, we review the related work in the field of deterministic analysis. In this context, Network Calculus [5,2] provides a useful approach, based on service curves, to obtain deterministic bounds. We evaluate this technique for the present problem in Sect. 3. In Sect. 4, we present and discuss an upper bound on the end-to-end delay. The related additive property is studied in Sect. 5. In Sect. 6 and 7, we generalize the results of [10] obtained in the two-server case. In Sect. 8, we further generalize these results to cover the case of general accumulation networks. Conclusion and hints for future works are eventually given in Sect. 9.

## 2 Related Work

The FIFO discipline is not able to offer a differentiated service. This is why deterministic studies have focused mainly on the design and evaluation of new service disciplines such as Packet Generalized Processor Sharing [6] which provides end-to-end delay bounds for leaky bucket sources.

Cruz [5] provides results concerning the burstiness characterization of flows inside a FIFO network. However, the bounds derived are not tight enough since they are obtained through summation of local worst-cases. The service curve and network service curve paradigms (see Sect. 3) enable us to obtain tighter bounds. A recent and major work in the field of FIFO networks is [3]. The authors show that if the peak rates of sources in a general FIFO network are constrained by a certain value (related to the number of flows that the source meets on its path), then the network is stable and bounds on end-to-end delays and backlogs are obtained. This is a major result since it applies to FIFO networks with a general topology, but with strictly deterministic sources. By contrast, we focus, in this work, on variable bit rate sources but for the more restrictive class of accumulation networks.

## 3 A Service Curve Approach

Network Calculus provides a straightforward way to model sources and network elements, through an arrival and a service curve respectively. It formulates theorems to derive bounds on backlog and delays. The arrival curve of a source represents, intuitively, an upper bound on the amount of traffic the source can send on any time interval. The service curve represents a lower bound on the service the source may expect.

A given flow crossing  $n$  network elements offering  $(\beta_i)_{i \in \{1,n\}}$  as service curves, may consider the network as a unique element with a (network) service curve which is the convolution of the individual service curves. The interest of the network service curve is that it provides a tighter bound on the end-to-end delay than summing the local bound that may be obtained at each server. To apply

this technique in the case of FIFO networks, a first step is to derive the service curve received by a given flow sharing a FIFO server with another flow, as illustrated next.

### 3.1 Service Curve in a FIFO Environment

Consider 2 sources,  $S_1$  and  $S_2$  (constrained by  $\alpha_1$  and  $\alpha_2$ ) and one FIFO server of capacity  $C$ . Let  $\lambda_C$  be the function such that :  $\forall t \geq 0$ ,  $\lambda_C(t) = Ct$ , and  $R_i$  (resp.  $R_i^*$ ) the cumulative rate function of  $S_i$  at the input ( resp. output) of the server. For a given time  $t$ , let us denote  $s_0$  the last time there was no backlog ( $s_0 \leq t$ ). Thus,  $R_1^*(s_0) = R_1(s_0)$  and  $R_2^*(s_0) = R_2(s_0)$ . Since the server is work conserving, this yields :

$$R_1^*(t) - R_1^*(s_0) + R_2^*(t) - R_2^*(s_0) = C(t - s_0) . \quad (1)$$

Causality also implies that  $\forall t$ ,  $R_2^*(t) \leq R_2(t)$ . Thus :

$$R_2^*(t) - R_2^*(s_0) \leq R_2(t) - R_2(s_0) . \quad (2)$$

Now, since  $S_2$  is constrained by  $\alpha_2$  and the server rate, we obtain :

$$R_2^*(t) - R_2^*(s_0) \leq \min(C(t - s_0), \alpha_2(t - s_0)) . \quad (3)$$

Mixing equation (1) and (3), we obtain :

$$R_1^*(t) - R_1(s_0) \geq C(t - s_0) - \min(C(t - s_0), \alpha_2(t - s_0)) . \quad (4)$$

Let us define  $(x)^+$  as  $\max(0, x)$ . A service curve for  $S_1$  is thus  $\beta_1 = (\lambda_C - \alpha_2)^+$ .

### 3.2 Discussion

The service curve obtained is thus conservative. Indeed, if  $S_2$  were preemptive over  $S_1$ , the service curve would be the same since, in this case,  $S_1$  receives only the remaining capacity unused by  $S_2$ . Besides, assume that  $S_1$  and  $S_2$  transit in a second server where they mix with a third source. To derive a service curve for  $S_1$  in the second server, an arrival curve for  $S_2$  at the input of the second server is required. Network Calculus provides a way to derive this arrival curve from the arrival curve of  $S_2$  at the input of server 1 and its service curve in server 1. But since the service curve for  $S_2$  at server 1 is pessimistic, the arrival curve for  $S_2$  at the second server will also be pessimistic. Thus, the conservative aspect of the result increases with the size of the network. This approach leads inevitably to pessimistic results. For instance, consider a single server and assume  $S_1$  and  $S_2$  have the same traffic descriptor, namely  $(p, R, M)$ , where  $p$  is the peak rate of the source,  $R$  the leak rate of the bucket and  $M$  its depth. Then, the following relation exists between the bound on delay  $D_{SC}$  obtained with the service curve approach and the exact value of the maximum delay  $D_{max}$  :  $D_{max} = \frac{C-R}{C} D_{SC}$ . Thus, when  $R \rightarrow \frac{C}{2}$  (stability requires that  $C > 2R$ ),  $D_{SC} \rightarrow 2D_{max}$ .

The drawbacks of this method lead us to envisage a new approach presented in the next section. Note, however, that we have not proved that it was not possible to find a better service curve. It remains an open problem.

## 4 The Additive Bound

Consider first a tandem accumulation network with  $p$  servers and  $n$  sources. A superposition of leaky bucket constrained sources may be seen as a single multi-leaky bucket constrained source with arrival curve being the sum of the individual arrival curves. We can thus group the sources entering the network at a given node and consider a network with  $p$  servers and  $p$  sources.

In [10], we show that if the input of a server is leaky bucket constrained, so is the output. When applied to node  $j$  of the considered accumulation networks, we obtain that the source seen at the input of this node is multi-leaky bucket constrained. Application of Network Calculus gives that the worst-case source at node  $j$  (which generates the maximum local delay) is obtained when the multi-leaky bucket source is greedy (a leaky bucket constrained source is greedy when it emits its bits as soon as possible : it thus first emits its maximum burst size at its peak rate  $p$  and then emits at its mean rate  $R$ ). A recursion from server  $j$  to server 1 proves that if all the sources crossing node  $j$  are greedy and synchronous, i.e. they begin their emission at the same time, then the aggregated source at the input of node  $j$  is the worst-case source. We thus obtain the local maximum delay at node  $k \in [1, j]$  is achieved when all the sources entering the network before at node  $k \in [1, j]$  are greedy and synchronous. The sum of these maximum local delays provides an upper-bound on the maximum end-to-end delay in the system. We call it the Additive Bound. We have only considered tandem networks so far but the results also hold for a tree network : the maximum local delays are obtained when all the sources entering the network are greedy and synchronous. Since the aggregated source seen at the output of a subtree of the accumulation network is multi-leaky bucket constrained, we only have to test the accuracy of the Additive Bound for tandem accumulation networks.

## 5 Additivity Property and Networks Classification

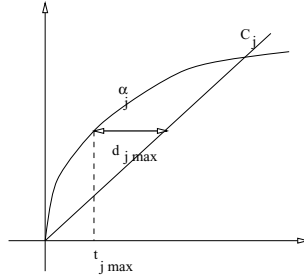
### 5.1 Definition

We term Additivity the following property : “There exists a trajectory of the system such that a bit experiences an end-to-end delay equal to the Additive Bound”. By extension, we say that a network is additive if it exhibits the Additivity property. It is non-additive otherwise.

### 5.2 Intrinsic Parameters ( $t_{max}, d_{max}$ ):

Let us denote greedy trajectory, the trajectory of the system where all the sources are greedy and synchronous. This yields the maximum delay on each node (see

Sect. 4). For this greedy trajectory and for each node  $j$ , we use the following parameters :  $d_{j \max}$ , the maximum local delay and  $t_{j \max}$ , the time where the bit experiencing  $d_{j \max}$  arrives at node  $j$  (see Fig. 1).



**Fig. 1.** Intrinsic Parameters

### 5.3 Networks Typology

In [10], we proved that a two-stage network is additive if and only if  $t_{2 \max} \leq t_{1 \max} + d_{1 \max}$ . This condition can be verified easily with the greedy trajectory of the system. We also characterize the “additive trajectory” :  $S_1$  and  $S_2$  greedy respectively from times  $\theta_1 = 0$  and  $\theta_2 = t_{1 \max} + d_{1 \max} - t_{2 \max}$ .

By extension, any general tandem accumulation network can be partitioned in a set of subnetworks for which the following property either holds or not :

**Property 1.** For all adjacent servers  $j$  and  $j+1$ ,  $t_{(j+1) \max} \leq t_{j \max} + d_{j \max}$ .

## 6 Additive Networks

We generalize here the result obtained for the two-server case. Consider an accumulation network with  $p$  servers for which Prop. 1 holds. Let us define  $(\theta_j)_{j \in \{1, p\}}$  as follow :

1.  $\theta_1 = 0$
2.  $\theta_{j+1} = \theta_j + (t_{j \max} + d_{j \max} - t_{(j+1) \max})$ ,  $j \in [1, p-1]$

If  $S_j$  is greedy from  $t = \theta_j$ , (note that  $\theta_{j+1} \geq \theta_j$ ), the bit experiencing  $d_{1 \max}$  in the first server experiences  $d_{j \max}$  at node  $j$  for all  $j \in \{1, p\}$ . Thus its end-to-end delay is :  $D_{\max} = \sum_{j=1}^p d_{j \max}$ . An accumulation network for which Prop. 1 holds is thus additive. Besides, since the only way for a bit to experience  $\sum_{j=1}^p d_{j \max}$  is to experience  $d_{j \max}$  at server  $j$ , for all  $j \in \{1, p\}$ , it follows that a network that does not fulfill Property 1 is not additive.

## 7 Non-additive Networks

In this section, we generalize the lower bound building method developed for a two-server network to accumulation networks of any size and use it to test the accuracy of the Additive Bound in the case of non-additive networks. A direct generalization would hide the difficulty of the construction of the trajectory. We thus first present the three-server case.

### 7.1 Three-Server Case

**Lower Bound.** Consider a two-stage network. If it is non-additive, this means intuitively that the burst necessary to obtain  $d_{1 \max}$  is not sufficient to obtain  $d_{2 \max}$  in the second server (considering the greedy trajectory of the system), since when all the bits of this burst have reached the second server, the delay that can be obtained is less than  $d_{2 \max}$ . The idea is then to postpone the emission of this burst so as to synchronize the local maximum delays, knowing that the delay on the second server will necessarily be less than the delay in the greedy synchronous case.

Consider now a network with three servers and three sources  $(S_i)_{i \in \{1, \dots, 3\}}$  ( $S_i$  entering at node  $i$ ). The trajectories of the sources are built so as to maximize the amount of bits in buffer  $j$  when the reference bit, i.e. the one experiencing  $d_{1 \max}$  in the first server, arrives.

**Trajectory of Sources.** Consider the greedy trajectory of  $S_1$ , as given in Fig. 2. It can be divided into three parts. The first part corresponds to the part of the trajectory necessary to achieve the local maximum delay  $d_{1 \max}$ . The second part corresponds to the time necessary for the last bucket of the sources composing  $S_1$  to empty. In the last part, all the sources composing  $S_1$  have reached their mean rate.

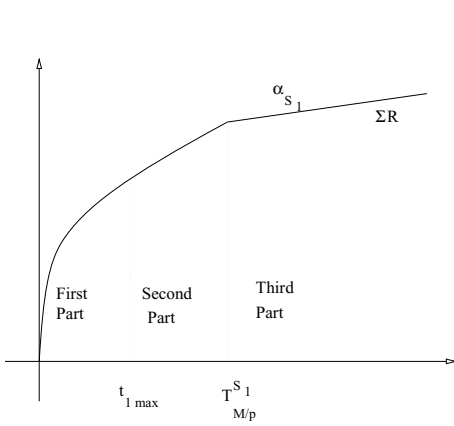
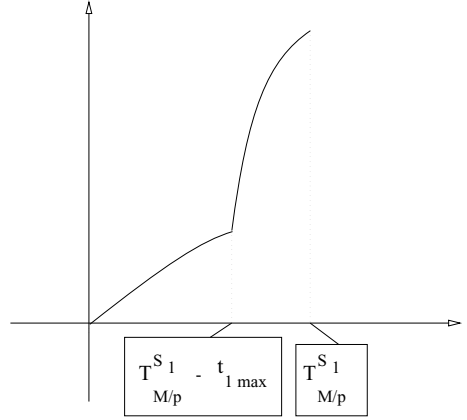
Now consider the trajectory of  $S_1$  given in Fig. 3.  $S_1$  is an aggregation of  $n_1$  sources, with each source controlled by a single leaky bucket with parameters  $(p_k, R_k, M_k)$  for  $k \in \{1, \dots, n_1\}$ . With the greedy trajectory of the system, the source with index  $k$  emits at its peak rate  $p_k$  during  $[0, \frac{M_k}{p_k}]$  and then emits at its mean rate  $R_k$ . Let us define :

$$T_{\frac{M}{p}}^{S_1} = \max_{k \in \{1, \dots, n_{S_1}\}} \left( \frac{M_k}{p_k} \right) . \quad (5)$$

$T_{\frac{M}{p}}^{S_1}$  corresponds to the beginning of the third part defined in Fig. 2. The modified trajectory is built by changing the beginning of emission of the sources composing  $S_1$  as follows :

1. if  $\frac{M_k}{p_k} \leq t_{1 \max}$  then the source :
  - (a) emits at its mean rate during  $[0, T_{\frac{M}{p}}^{S_1} - t_{1 \max}]$ ,
  - (b) becomes greedy for  $t \geq T_{\frac{M}{p}}^{S_1} - t_{1 \max}$ ; this is possible since its bucket is still full at this time.



Fig. 2.  $S_1$  initial trajectoryFig. 3.  $S_1$  modified trajectory

2. if  $\frac{M_k}{p_k} \geq t_{l max}$ , then the source :
  - (a) emits at its mean rate during  $[0, T_{M/p}^{S1} - \frac{M_k}{p_k}]$ ,
  - (b) becomes greedy for  $t \geq T_{M/p}^{S1} - \frac{M_k}{p_k}$ .

The modified trajectory has two parts (see Fig. 3) :

1. the first part where some sources emit at their peak rate whereas others emit at their mean rate. This part corresponds to the second part of the initial greedy trajectory with a slight modification : if a source emits at its peak rate during  $\tau_1$  and then at its mean rate during  $\tau_2$  in the initial trajectory, then, in the modified trajectory, it first emits at its mean rate during  $\tau_2$  and then at its peak rate during  $\tau_1$ . Due to this inversion between  $\tau_1$  and  $\tau_2$ , we call inverted part this part of modified trajectory.
2. the second part is strictly equivalent to the first part of the initial trajectory.

Note that, as with the initial trajectory, the last bucket empties at time  $t = T_{M/p}^{S1}$ .

A modified trajectory for  $S_2$  and  $S_3$  is built using the same method. We now fix the synchronization parameters.

**Synchronization of Sources.** With the modified trajectory given above for  $S_1$ , the last bit of the burst (reference bit) experiences a delay  $d_{l max}$ .  $S_2$  is synchronized in such a way that the end of its burst corresponds to the arrival of the reference bit. This bit will then experience  $d_2 \leq d_{l max}$  in the second server. Since, a priori,  $T_{M/p}^{S1} \neq T_{M/p}^{S2}$ , using the previous synchronization method leads one of the sources to start emitting before the other. Assume  $S_2$  starts its emission before  $S_1$ . To maximize the number of bits backlogged in server 2 at the time where the reference bit arrives, it is possible to modify the trajectory

of  $S_1$  such that it emits at its mean rate before the beginning of the modified trajectory, in an interval of length  $T_{\frac{M}{p}}^{S_2} - T_{\frac{M}{p}}^{S_1}$ . This trajectory of  $S_1$  is valid with respect to its leaky bucket constraint. The same method is applied to synchronize  $S_3$ , as shown in Fig. 4.

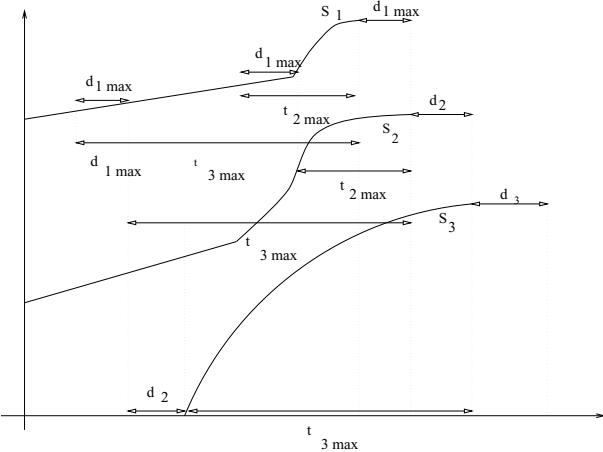


Fig. 4. Sources synchronization

**Result for Delay.** The lower bound on the maximum end-to-end delay is obtained as the end-to-end delay of the reference bit in the modified trajectory. Since all the sources are leaky bucket constrained, the initial and modified trajectories described above correspond to piece-wise linear curves. Computation of the intrinsic parameters as well as the delay of the reference bit is thus straightforward from the algorithmic point of view.

7.2 Numerical Results

Our aim is to estimate the accuracy of the Additive Bound in a non-additive network with the lower bound presented above. Accuracy means here the relative difference between the Additive Bound and this lower bound. A key problem is the choice of the non-additive networks.

Networks Building Method.

We consider accumulation networks with  $p = \{4, 5, 8, 10\}$  servers. For each server, we draw the number of sources entering at this stage in a uniform fashion in the interval  $[1, 5]$ . Then, we draw the characteristics of the sources in Table 7.2. We now have to choose the capacities of the servers. A necessary condition

**Table 1.** Sources descriptors

<i>Peak rate p</i>	<i>Mean rate M</i>	<i>Burstiness M</i>
10	0.1	10
100	1	100
1000	10	1000

for a network to be additive is that the rate of the servers increases. Conversely, if capacities decrease, the network is non-additive (sufficient but not necessary). We choose to give every server the same rate which is the sum of the mean rate of all the sources times  $\gamma = 1.01$  ( $\gamma$  is used to ensure stability). This sum represents the minimum capacity of the last server in the case of accumulation networks . Doing so, the most important part of the end-to-end delay is concentrated at the end of the network. To obtain some significant results, we choose to calculate the relative range, which is the difference between the lower bound and the Additive Bound divided by the Additive Bound, for this initial system, i.e. a particular random generation of the sources descriptors and capacities of servers. We next modify the network by enforcing some of the sources to modify their entering node. This is achieved through the following algorithm :

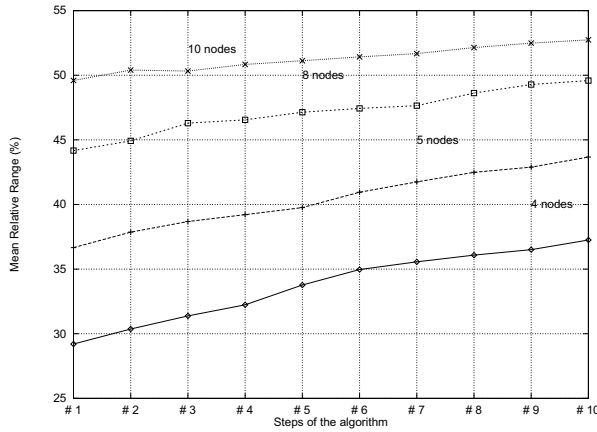
1. Step 1 : one computes the relative range for the initial network.
2. Step 2-9 : each source is removed from node  $j$  to node  $j - 1$  with probability 0.1.

Applying this algorithm, the accumulation network heuristically “worsens” and thus the relative range should increase.

**Results.** The results, presented in Fig. 5, are obtained for 10000 successive random generations of networks. The x-axis is indexed following the steps of the algorithm. For each step and for the different network sizes, we compute the mean relative range.

**Discussion.** For non-additive accumulation networks, we have an upper bound on the end-to-end delay, the Additive Bound, and a heuristically obtained lower bound. The maximum, exact, end-to-end delay over all possible trajectories of the system is thus between these two bounds and gives full meaning for considering the relative range as a performance parameter.

The results obtained confirm the good accuracy of the lower bound. The mean relative ranges remain reasonable even for large size of networks. The maximum error, not presented here is no more than 67%. It thus remains within the same order of magnitude. We now address the case of general accumulation networks which are neither strictly additive nor strictly non-additive.



**Fig. 5.** Results with network of different sizes

## 8 General Accumulation Networks

Our expectation is that the Additive Bound always represents an accurate upper bound on the maximum end-to-end delay for accumulation networks. Proving such a statement requires an exhaustive study, which is not possible. We restrict our study to a specific class of accumulation networks, that we term well-formed accumulation networks. A well-formed accumulation network is an accumulation network where the following rule applies : capacities of the servers follow an increasing fashion from the leaves to the root of the tree.

We extend here the previous results for well-formed tandem accumulation networks using the same lower bound as in the non-additive case. Indeed, the method used to build the trajectory leading to the lower bound is based solely on the set of intrinsic parameters  $(t_{j \max}, d_{j \max})$  and do not rely on any assumption concerning the additivity of the network. It may thus be applied in the case of well-formed accumulation networks.

### 8.1 Networks Building Method

The method used to generate a well-formed network is the following :

1. for each server, the number of sources (between 1 and 5) entering at this node and their characteristics are drawn (see Table 7.2).
2. the rate of each server is then computed as the sum of the mean rate of the sources crossing this node times a coefficient  $\alpha$ .  $\alpha$  can take one of the three following values  $\{1.1, 1.5, 2.0\}$  which, for each set of sources, leads to three different networks.

## 8.2 Results

We present hereafter the numerical results obtained for networks of various sizes (from 3 to 20 servers). For each network size, 10000 networks are drawn. The performance parameter computed for each network is the relative range between the lower bound and the Additive Bound. The average relative ranges are presented in Table 8.2.

**Table 2.** Average relative ranges (in %) with networks of different sizes

	<i>Size=3</i>	<i>Size=5</i>	<i>Size=10</i>	<i>Size=15</i>	<i>Size=20</i>
$\alpha = 1.1$	0.72	1.29	2.03	2.89	3.87
$\alpha = 1.5$	1.97	3.36	5.67	8.42	11.23
$\alpha = 2.0$	1.96	3.13	5.06	7.64	10.18

## 8.3 Discussion

The results shown in Table 8.2 strongly confirm our claim : the Additive Bound represents a good criterion for a deterministic CAC (a CAC based on a deterministic delay bound). They are also interesting since the way well-formed accumulation networks are built here is close to a real dimensioning process. Indeed,  $\alpha^{-1}$ , which is the rate of the server divided by the sum of the average rates of the sources it serves, represents the average activity rate of the servers and tuning activity rates of servers at a given rate is a common procedure for networks dimensioning.

Compared to the results obtained in the previous section, the relative ranges obtained here are significantly smaller : for instance, for a network with 10 servers, the relative range was close to 50% whereas here it is close to 5%. This is due to the method the server rates are assigned in each case. In the previous section, all the servers had the same capacity which lead to a strictly non-additive network, whereas here, the rates increase from one server to another, which is a necessary (though not sufficient) condition to obtain additive networks.

## 9 Conclusion

In this paper, we focused on the problem of determining an end-to-end delay bound in an accumulation network. We first evaluate the direct service curve approach provided by the Network Calculus theory and stress the over-conservatism of the obtained results. We then propose a new approach, based on a trajectory analysis and the Additivity property introduced for a two-server case in [10]. We show that the Additive Bound (sum of the local maximum delays) represents a good approximation for the maximum end-to-end delay.

Future work will address the problem of designing a CAC procedure based on the Additive Bound. Preliminary results underline the problem of checking the non-violation of the QoS of the already established connections in a network environment. It seems that a heavy updating procedure needs to be performed after each new source acceptance. Extension of the network topologies is also an important point to consider. Up to now, we have addressed accumulation networks. We now wish to investigate the case of general networks topologies. However, the Additive Bound may be less accurate due to the fact that sources mixing with the reference source may leave this source before its exiting node. Preliminary results indicate that the computation of the local maximum delay is a very difficult task in a general FIFO network with leaky bucket constrained sources.

## References

1. H. Ahmed, R. Callon, A. Malis, and J. Moy. Ip switching for scalable ip services. *Proceedings of the IEEE*, 85(12), 1997.
2. J.-Y. Le Boudec. An application of network calculus to guaranteed service networks. *IEEE/ACM Transactions on Information Theory*, May 1998.
3. I. Chlamtac, A. Faragó, and H. Zhang. A deterministic approach to the end-to-end analysis of packet flows in connection-oriented networks. *IEEE Trans. on Networking*, August 1998.
4. D. D. Clark, S. Shenker, and L. Zhang. Supporting real-time applications in an integrated services packet network : Architecture and mechanism. *ACM SIGCOM'92*, 1992.
5. R.L. Cruz. A calculus for network delay, part 1 : network elements in isolation. *IEEE Transaction on Information Theory*, January 1991.
6. A. K. Parekh and R. G. Gallager. A generalized processor sharing approach to flow control in integrated services networks : The single-node case. *IEEE/ACM Transactions on Networking*, June 1993.
7. A. K. Parekh and R. G. Gallager. A generalized processor sharing approach to flow control in integrated services networks : The multiple-node case. *IEEE/ACM Transactions on Networking*, Mars 1994.
8. H. G. Perros and K. H. Elsayed. Call admission control schemes : A review. *IEEE Magazine on Communications*, Nov. 1996.
9. K. Shiimoto, N. Yamanaka, and T. Takahashi. Overview of measurement-based connection admission control methods in atm networks. *IEEE Communications Surveys*, 1999.
10. G. Urvoy, G. Hébuterne, and Y. Dallery. Delay-constrained vbr sources in a network environment. In *ITC'16*, 1999.

# Performance Analysis of Ip Switching and Tag Switching<sup>1</sup>

Gargi Banerjee, Rob Rosenberry, and Deepinder Sidhu

Maryland Center for Telecommunication Research  
University of Maryland Baltimore County  
1000 Hilltop Circle, Baltimore, MD 21250  
{gargi, rosenber, sidhu}@cs.umbc.edu  
Tel: 410-455-3063. Fax: 410-455-3964

**Abstract.** To cope with exponential growth of traffic on the Internet, IP networks require routers capable of much higher performance capabilities. This has led to interest in combining speed of layer-2 switching with scalability of layer-3 routing, leading to the generation of layer3 switches. Various technologies like IP, tag and Multi-Protocol Label Switching are being developed to provide these features by performing IP routing over ATM. In this paper we perform several experiments to analyze and compare the performance of IP switching and tag switching on networks comprising traditional IP routers running over an ATM backbone. Using network traces we compare the performance gains of IP switching to the gains obtained from tag switching. Our findings indicate that performance of IP switching is dependent on flow classification parameters and router speed. Also we find that there are higher performance gains with tag switching at the cost of moderately higher resource usage.

## 1. Introduction

The remarkable growth of the Internet has created a wealth of technical challenges. To meet the growing demand of bandwidth, Internet service providers (ISPs) need higher performance routing products. ATM has received much attention because of its high capacity, bandwidth scalability and its ability to support multi-service traffic. Thus, the need to evolve the routing functionality of the Internet and IP networks and the fact that ATM switches have the potential to provide great performance improvements over earlier network technologies have led to the desire to integrate IP and ATM. Several organizations have come up with different solutions for integrating IP network protocols into an ATM model. However, because of its connection oriented paradigm, the overlay model proposed by The ATM Forum for mapping

---

<sup>1</sup> This research was supported in part by the Maryland Center for Telecommunications Research (MCTR) at the University of Maryland, Baltimore County with funds received from the Department of Defense. The views and conclusions contained in this document are those of the authors and should not be interpreted as representing the official policies, either expressed or implied, of any sponsor of MCTR.

connectionless IP onto ATM has led to complex and inefficient implementations. The Internet Engineering Task Force (IETF) has advocated an alternative peer model. Several companies have published Request For Comments (RFC) describing various approaches to the technique known as IP label switching. Label switching technology enables IP control protocols to run directly on ATM hardware. ATM switches still use label-swapping mechanisms to forward cells but employ IP control protocols to set up forwarding of packets and allocate resources. This approach has the advantage of requiring only standard IP routing protocols and not any new signaling or address resolution protocols.

In 1996, Ipsilon published their label switching technology, which they called IP switching and claimed large performance improvements over traditional router [1]. Concurrent with Ipsilon's work on IP switching, engineers from Toshiba described their cell switch router [7], Cisco Systems announced tag switching and IBM published drafts on Aggregate Route-based IP Switching (ARIS). In 1997, the Internet Engineering Task Force (IETF) decided to create a standardized label switching technique from the incompatible offerings of the different vendors. Multi-Protocol Label Switching (MPLS) has been adopted as the standard name for layer3 switching technique [8].

This paper analyzes and compares the performance of the label switching techniques of IP and tag. Sections 2.1 and 2.2 give some background on the two techniques. Section 3 introduces the event driven simulator, which we developed to perform the analysis and comparisons for label switching techniques. In Section 4 we examine our simulations of the two techniques and compare important statistics. Finally, in Section 5, we present our conclusions about IP and tag switching.

## 2. Background

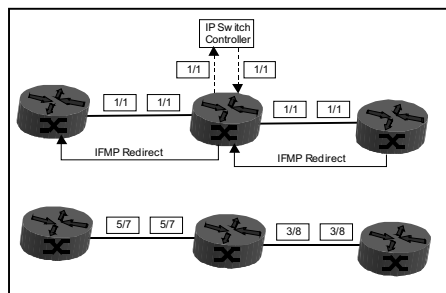
### 2.1 IP Switching

Ipsilon designed IP switching based on the belief that the peer model for mapping IP onto ATM was too complex as two address spaces, two topologies, and two routing protocols must be managed. Ipsilon also wanted IP switching to preserve the connectionless and stateless behavior of IP networks [2].

All IP switches contain two control protocols to manage the stream of IP flows: the Ipsilon Flow Management Protocol (IFMP) and the General Switch Management Protocol (GSMP). IP switches also contain an inherent flow classifier. This component identifies flows and decides whether they should be bound to ATM virtual channels for possible switching. Two types of flows are defined [2]:

1. Flow Type 1: a flow of IP datagrams between two applications, identified by the source and destination IP addresses along with the source and destination ports in the IP packet header.
2. Flow Type 2: a flow of IP datagrams between two hosts, identified by the source and destination IP addresses in the IP header.





**Fig. 1.** IP Switch Changeover from Routing to Switching

Fig. 1 illustrates the redirection of IP packets from routing to switching. Initially, all IP datagrams are sent across the default virtual channel 1/1 between two IP switches. At each IP switch, the packets are routed by traditional IP routing mechanisms. When IP switch B decides to create a switched path for a flow, it sends its upstream neighbor switch A an IFMP Redirect message asking to assign a specific vpi/vci value 5/7 for the flow. Now A can set up a switched connection to B for this flow. At this point, all packets belonged to the selected flow are switched across the newly established connection instead of routed over that hop [3]. Similarly B's downstream neighbor C for this flow also sends a Redirect to B assigning a label 3/8 to the flow. Thus now a switched path is created for the flow from B to C. IFMP is a soft state protocol and the installed state at a switch needs to be updated periodically.

GSMP is a simple, generic switch management and control protocol designed to manage resource allocation within an IP switch. It is solely used internally within an IP switch to communicate switching commands to the controller for the ATM switch fabric. GSMP messages provide connection management, port management, statistics, configuration, and event notification functions [7].

## 2.2 Tag Switching

The major difference of Cisco's tag switching from IP switching is that while IP is a data driven approach, tag switching is control-driven [7]. A router that supports tag switching is called a Tag Switching Router (TSR). Tag switching consists of two components: forwarding and control. The forwarding component uses tag information carried by packets to perform forwarding. Every tag switch has a Tag Information Base (TIB). Each entry in the TIB contains the incoming tag and one or more subentries of the form outgoing tag, next hop, and outgoing interface [4]. When a packet with a tag is received by a TSR, the switch uses the tag as an index in its TIB. If it finds an entry with the incoming tag equal to the tag carried in the packet, then it replaces the tag in the packet with the outgoing tag. It also replaces the link level information in the packet such as a MAC address with the outgoing link level information, and forwards the packet over the outgoing interface. [4]

The control component is responsible for creating tag bindings, and then distributing that information among the tag switches in the network. Distribution of the tags may be carried out by piggybacking binding information on existing routing

protocols or by using a separate Tag Distribution Protocol (TDP). There are three methods for tag allocation and TIB management:

1. Downstream tag allocation: the tag carried in a packet is generated and bound to a routing prefix by the tag switch at the downstream end of the link with respect to the direction of data flow.
2. Downstream tag allocation on demand: tags are only allocated and distributed by the downstream tag switch when it is requested to do so by the upstream tag switch.
3. Upstream tag allocation: a tag switch creates tag bindings for outgoing tags and receives bindings for incoming tags from its neighbors.

### 2.3 Multi-protocol Label Switching (MPLS)

MPLS was proposed by IETF as the standard for label switching technology. Since it combines layer three routing with layer two switching it is said to lie at layer two-and-a-half in the network stack. The MPLS label distribution protocol (LDP) operates in two modes- downstream on-demand distribution and downstream unsolicited distribution. Even though it resembles tag switching closely, MPLS has today evolved as one of the most powerful traffic engineering tools for next generation networks. LDP can also be extended to perform constraint-based routing (CR-LDP), where constraints may be an explicit route or a policy based route. Thus MPLS promises to have a distinct role in providing quality of service across the Internet.

## 3. Label Switching Simulator

The simulator that we designed and implemented for the analysis of label switching techniques using ATM is an extension of the NIST ATM/HFC Network Simulator [12]. Our discrete event simulator models an IP label switching network using existing components such as B-TEs, ATM switches, and high speed links. We have designed label switching specific components like IP switches, tag edge routers, and tag switch routers. The IP switch component models the operation of the IFMP protocol. It does not implement the GSMP protocol as that protocol operates entirely within the IP switch. Both type1 and type2 flows may be recognized by the switch. The flow classification algorithm, which we have implemented, uses a static X/Y classifier [1]. For our simulations, we have used values of 5 packets for X and 10 seconds for Y with a 30-second timeout. The tag edge router component for the simulator is the router at the edge of the tag switched network. It receives untagged packets from the traffic generator and tags them with labels from its TIB. It also participates in TDP by sending out BIND\_REQUESTS to its neighbors for each route entry in its FIB. When it receives BIND\_INFO messages from other TSRs, it makes an entry in its TIB for that route. Since our simulator is based upon an ATM network, the tag edge router also performs the added function of segmentation and reassembly of IP packets into ATM cells using AAL5.

4. Performance of Label Switching Techniques

In order to test the performance of our simulated label switches, we obtained IP trace files from the Association of Computing Machinery’s Internet Traffic Archive. These files were collected from real traffic flowing into and out of the Lawrence Berkeley Laboratory in 1994 and Digital Equipment Corporation in 1995. [1] Indicates that the ‘DEC1’ and ‘DEC2’ files contain 2,983,221 and 3,467,733 packets respectively with an average of over 800 packets per second. ‘DEC1’ has 75,438 Ipsilon type 1 flows and 45,560 type 2 flows. The ‘DEC2’ file has 75,739 type 1 flows and 37,252 type 2 flows.

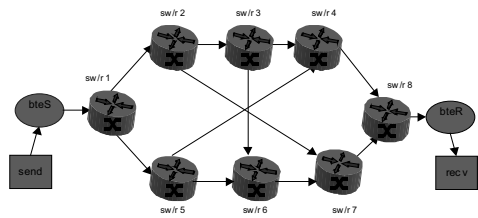


Fig. 2. Network of 8 switches

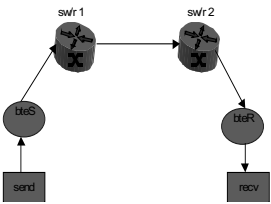


Fig. 3. Network of 2 switches

Fig. 2 and 3 illustrate the label switched networks that we used in our simulations. The ‘send’ component is the traffic generator for the simulations. It is responsible for injecting the packets from the trace file into the network at the appropriate time based upon a timestamp attached to each packet. The ‘bteS’ and ‘bteR’ components are artifacts of the design of the simulator and not relevant to the performance of this simulation. The ‘SW/R’ components are the label switch routers. They are either IP switches or a combination of tag edge routers and tag switch routers depending upon the label switching technique being simulated. The ‘recv’ component receives packets sent through the network.

4.1 Single Switch Performance

In the first experiment we verified the IP switching results as presented in [1]. Since these results were obtained for a single switch, we also used the network configuration of Fig. 3 where cells were switched only at a switch1. The trace files used in this experiment had also been used in [1].

Figs 4.1a-b shows the percentage of datagrams switched for both type1 and type2 flows using all the corporate traces. The results show that about 80-85% of datagrams is switched for all the traces. Fig. 4.1c shows how larger flow creation delays decrease percentage of switched datagrams. Fig. 4.1d indicates that a flow deletion delay of 50-60secs achieve a switching percentage of 75-80% for both type1 and type2 flows. All these results were found to be similar to results observed in [1].

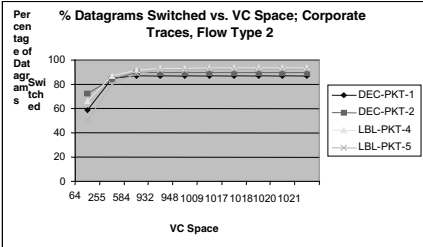


Fig. 4.1a. Effect on VC space (type2)

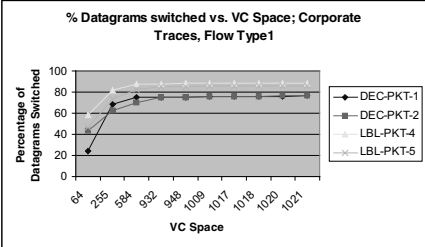


Fig. 4.1b. Effect on VC space (type1)

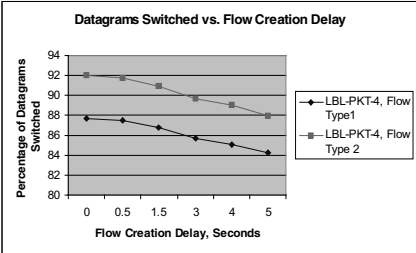


Fig. 4.1c. Effect of flow creation delay

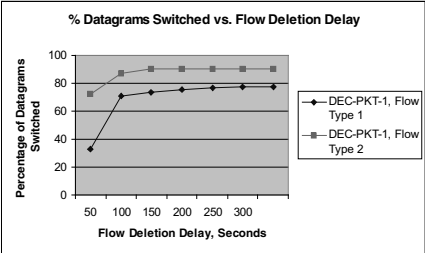


Fig. 4.1d. Effect of flow deletion delay

4.2 Effect of Router Speed on Switching Performance

In this experiment we investigate the effects on routing speed on IP and Tag switching respectively. The network configuration of Fig. 2 is used for this experiment. Only type1 flows are considered. Router processing time is the time taken by a router to process each incoming datagram. This mainly includes delay due to re-assembly of ATM AAL5 cells into an IP datagram, layer 3 routing lookup based on destination address in the IP header as well as flow classification decisions, if necessary and fragmentation of IP datagram back into ATM AAL5 cells. The input and output buffer delays are not considered as part of the router processing delay. For data-driven switching approaches, the percentage of switched datagrams is largely dependent on flow classification routines, and hence on the second factor in the delay component. We used two values of routing delay for this experiment: 100 ms and ms.

Fig. 4.2a and Fig. 4.2c show the percent of packets that are switched instead of routed at the edge switch and a representative interior switch respectively. The performance of the interior IP switch drops dramatically from over 80% packets switched to fewer than 20% while the performance of the tag switch remains constant. Fig. 4.2b and Fig. 4.2d show the performance of the same label switches using values of 1 millisecond for the routing delay and 10 microseconds for the switching delay at each of the label switches. In this case, the operation of the interior IP switch remains above 80% packets switched while again the tag switches maintain constant 100%

packets switched. The results showed that while for tag switching the percentage of switched datagrams remained the same, for IP the performance gain obtained from the percentage of switched datagrams reduced considerably with the large routing delay. This is due to the fact that IP being a data-driven approach, flow classification is done as and when data comes in. Since a large routing delay, implies a slow flow-classification module, hence most of the datagrams are already sent to route-buffers before a switching decision can be taken. Once the packets reach route buffers they have to be routed, and hence the switched percentage drops. However in a control driven approach like Tag, once the tags are set up, all cells may be switched and there is no need for any routing decisions. Hence the tag switching performance remains largely independent of the large router processing delay.

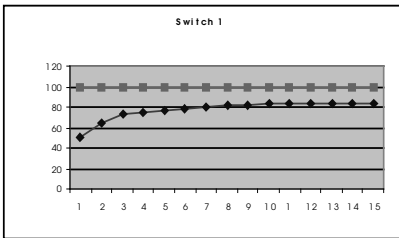


Fig. 4.2a. Edge Switch–100ms delay

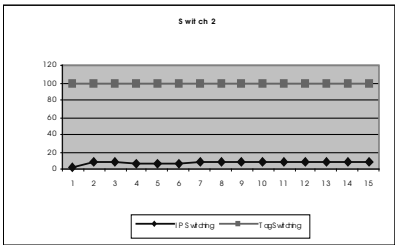


Fig. 4.2c. Interior Switch-100ms delay

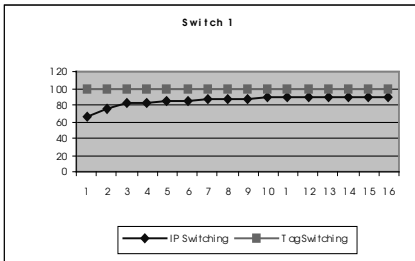


Fig. 4.2b. Edge Switch–1ms delay

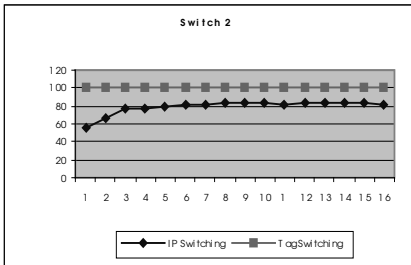


Fig. 4.2d. Interior Switch–1ms delay

4.3 Type1 vs. Type2 Classification

In this experiment we investigated the performance of two different types of flow classification for IP switching using the DEC-PKT1 trace file. The configuration of Fig. 2 is used for this experiment.

From the graphs 4.3a-e we see that flow type 2 achieves a higher switching percentage, a lower VC usage and a marginally lower average packet delay, when compared to flow type 1. The drawback though is that with flow type 2, due to larger number of datagrams being sent on the same circuit, delays due to network congestion may offset some of the benefits of having a type2 classification.

4.4 IP Switching vs. Tag Switching

In this experiment we compared the performance gains obtained from tag switching versus those obtained from IP switching. All the runs in this experiment were for trace files DEC-1and DEC-2. The network configuration of Fig. 2 is used for this experiment. Only type1 flows are considered. The comparison was based on the percentage of cells and packets switched VC space usage, message overhead and average packet delay.

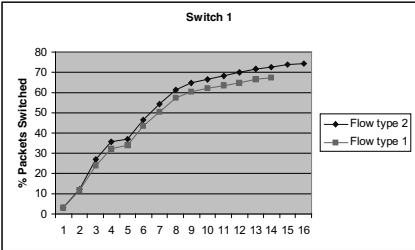


Fig. 4.3a. % packets switched at switch1

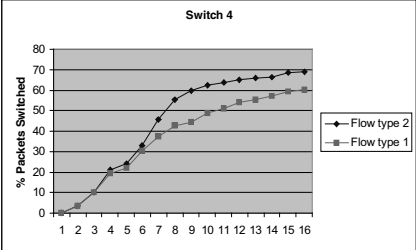


Fig. 4.3b. %packets switched at switch4

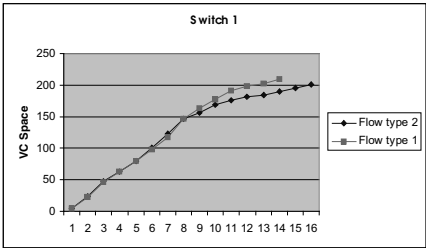


Fig. 4.3c. VC space usage at switch1

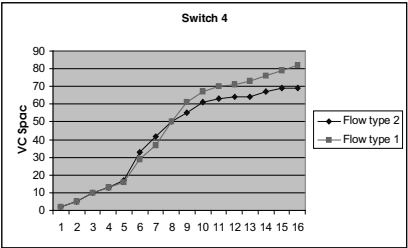


Fig. 4.3d. VC space usage at switch4

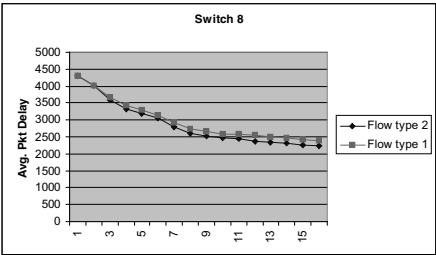


Fig. 4.3e. Average Packet delay at egress switch8

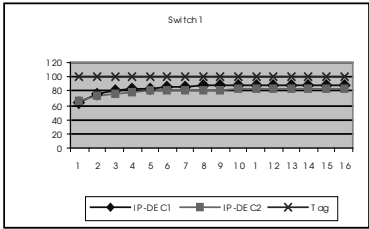


Fig. 4.4a. Switching at switch1

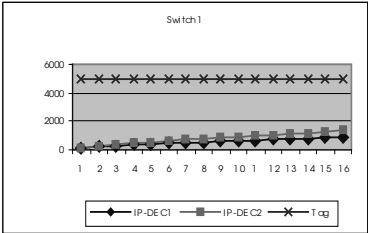


Fig. 4.4c. VC-space usage at switch1

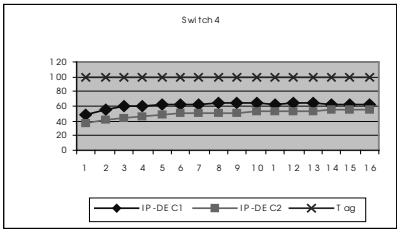


Fig. 4.4b. Switching at switch4

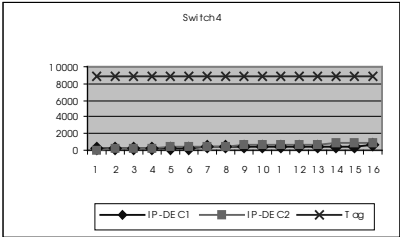


Fig. 4.4d. VC-space usage at switch4

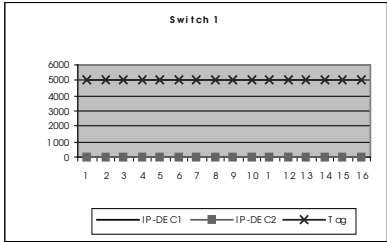


Fig. 4.4e. Message Overhead at switch1

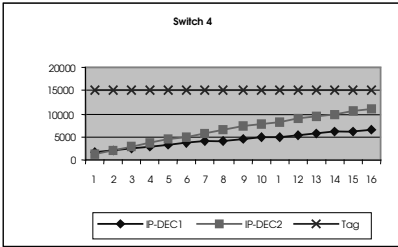


Fig. 4.4f. Message Overhead at switch4

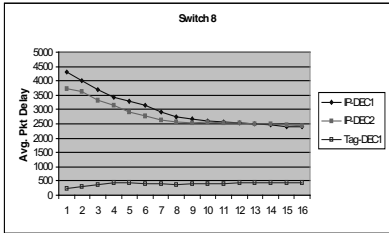


Fig. 4.4g. Average Packet Delay at egress switch8

Fig. 4.4a, b graphs the performance of the ingress (switch 1) and a representative interior switch (switch 4) in our simulated network for the two DEC trace files. Similar results were noted for the other interior switches and for other trace files. The traces on each graph show how each label switching technique performed against the three IP trace files that we used. Only the DEC-1 trace is drawn for the tag switching method as all of the trace files performed identically. The IP switching technique performed differently for each file since the number and length of flows changed from one trace file to another. The graphs show that the tag switching technique performs consistently against any type of traffic patterns, as it is a control driven label-switching technique. All switched paths are pre-established based upon the routes in the forwarding databases of the switches. Since the IP switching technique is data driven, it is more sensitive to variances in the data flows. The ingress and egress switches achieve approximately 80% packet switching while the interior switches only manage from 40% to 70% switching of packets. The tag switching technique maintains a very high amount of VC space utilization as shown in 4.4c, d. One virtual channel is allocated for every route path through the switch. The VC space used remains constant since our route tables are static. For the IP switching technique, the VC space utilized begins at zero and grows as the number of switched flows increases. However, since a VC is released as the traffic in the flow diminishes, the total number of VCs in use at any point remains much lower than with the tag switching method. Fig.4.4e, f represents the message overhead incurred to establish and maintain switched traffic throughout the network. The amount of overhead traffic is shown to directly relate to the amount of VC space utilized by the label switching technique. Since the tag switching mechanism uses more virtual channels, its message overhead is also correspondingly higher. Fig.4.4g shows the average packet delay across the network. This parameter measures the average end to end delay encountered by a packet and is measured at sample intervals spaced 1 sec apart. From the measurements taken during our simulations in experiment4, we conclude that the tag switching technique results in higher performance across the network. For a five to ten-fold increase in the number of virtual channels used, we gain a 20% to 40% performance increase. With today's ATM switching equipment, an increased VC utilization translates into higher memory requirements for the switch table and VC control storage. The switch table lookup hardware must also be able to handle searching through larger tables. However, we believe that this approach is much more feasible than trying to increase the route processing hardware. Trying to improve either software based routing system or hardware based routing mechanism is more complicated and difficult to achieve than improving the switching components.

## 5. Conclusions

In this section, we present our conclusions based on the results obtained from our simulations of two label switching techniques. First we have succeeded in validating results of [1] which mentions an overall 80% switching in IP switches. We have also



shown that switching performance of a data driven approach depends largely on the routing capacity of the layer 3 switches. A slow router with a processing time in the order of 100ms can to a large degree offset the advantages obtained from label switching. In the third experiment we have seen that if quality of service is not so much of a concern, then IP switching with type 2 classification achieves high level of switched datagrams with a lower channel usage. Finally, we compared the performance of Cisco Systems' tag switching and Ipsilon's IP switching methods. The tag switching technique delivers higher performance in percent of packets switched at a higher cost of more virtual channels used and increased message overhead. We believe that the higher performance gained with the tag switching system are worth the associated costs.

As future work, our IP Switching and Tag Switching studies can be extended to Multi-protocol Label Switching (MPLS) which is being proposed as an IETF standard for the Internet. The issues to be investigated are: network throughput, latencies due to Label Distribution Protocol (LDP) and its implications, mapping MPLS onto ATM Quality of Services, traffic engineering using MPLS labels, support for multicast with MPLS and security in MPLS networks.

## References

- [1] Steven Lin, Nick McKeown. „ A Simulation Study of IP Switching.“ *Proc. ACM SIGCOMM '97*, France, Sept. 1997.
- [2] Peter Newman, Tom Lyon, Greg Minshall. „ Flow Labelled IP: A Connectionless Approach to ATM. *Proc. IEEE Infocom '96*, vol. 3, pp.1251-60.
- [3] P. Newman, W.L. Edwards, R. Hinden, E. Hoffman, F. Ching Liaw, T. Lyon, G. Minshall. „ Ipsilon Flow Management Protocol Specification for IPv4.“ *IETF RFC 1954*, May 1996.
- [4] Yakov Rekhter, Bruce Davie, Dave Katz, Eric Rosen, George Swallow. „ Cisco Systems' Tag Switching Architecture Overview.“ *IETF RFC 2105*, Feb. 1997.
- [5] Hao Che, San-qi Li, Arthur Lin. „ Adaptive Resource Management for Flow-Based IP/ATM Hybrid Switching Systems.“ *IEEE/ACM Transactions on Networking*. Vol. 6, No. 5, Oct. 1998.
- [6] R. Cole, D. Shur, C. Villamizar. „ IP over ATM: A framework document.“ *IETF RFC 1932*, Apr. 1996.
- [7] Bruce Davie, Paul Doolan, Yakov Rekhter, "Switching in IP Networks: IP Switching, Tag Switching, & Related Technologies", 1998.
- [8] R. Callon, P. Doolan, N. Feldman, A. Fredette, G. Swallow, A. Viswanathan, "A Framework for Multiprotocol Label Switching", Work in Progress, July, 1999.
- [9] E. Rosen, A. Viswanathan, Callon, "Multiprotocol Label Switching Architecture", Work in Progress.'98
- [10] B. Davie, J. Lawrence, K. McCloghrie, Y. Rekhter, E. Rosen, G. Swallow, P. Doolan, "Use of Label Switching With ATM", Work in Progress, September, 1998.
- [11] Juha Heinanen, " Multiprotocol Encapsulation over ATM Adaptation Layer 5." *IETF RFC 1483*, July 1993.
- [12] NIST, „ The Nist Simulator ATM/HFC Network Simulator. Operation and Programming.“

# A distributed Mechanism for Identification and Discrimination of non TCP-friendly Flows in the Internet

Thomas Ziegler<sup>1,2</sup>, Serge Fdida<sup>1 \*</sup>

<sup>1</sup> Université Pierre et Marie Curie, Laboratoire Paris 6; 75015 Paris, France

{Thomas.Ziegler, Serge.Fdida}@lip6.fr

<sup>2</sup> Polytechnic University Salzburg, 5020 Salzburg, Austria

Thomas.Ziegler@newmedia.at

**Abstract.** This paper proposes the MUV (Misbehaving User Vanguard) algorithm for identification and discrimination of non TCP-friendly best-effort flows. The operational principle of MUV is to detect non TCP-friendly flows at the ingress router by comparing arrival rates to equivalent TCP-friendly rates, i.e. the arrival rate of a TCP flow having the same round-trip time and packet-loss probability. If a flow is identified as non TCP-friendly, its packets are marked as “unfriendly”. Core routers discriminate packets marked as unfriendly with RED-based drop-preference mechanisms. In order to measure the round-trip time and the packet-loss probability for the computation of a flow’s TCP-friendly rate, ingress router and egress router communicate via a simple protocol.

The MUV algorithm fits into the Differentiated Services Architecture of the Internet and can be considered scalable as it only requires per-flow state at ingress- and egress routers. We show by simulation that MUV is able to reliably identify and discriminate unresponsive flows and investigate its performance bounds regarding the identification of flows using non TCP-friendly congestion control algorithms.

## 1 Introduction

With the emergence of multimedia- and real-time applications today’s Internet traffic is no longer solely TCP but a mix of various applications and transport protocols utilizing heterogeneous congestion control algorithms. This heterogeneity of congestion control causes considerable fairness problems in the Internet. In particular, so-called “unfriendly flows” reducing their transmission rates less conservatively in response to congestion indications from the net (i.e. packet loss) than “friendly flows” tend to grab an unfairly-high share of bandwidth and bufferspace. As a result, Internet users have an incentive to be misbehaving, to utilize non-conservative congestion control mechanisms and thereby generate flows having a low level of friendliness<sup>1</sup> [1][2].

“Unresponsive flows” have no end-to-end congestion control implemented at all hence they do not back off in response to packet loss. Simulations in [2] show extreme unfairness among a friendly TCP and an unresponsive CBR flow<sup>2</sup> sharing a link. The throughput of the TCP flow converges to zero while the throughput of the CBR flow converges to the link bandwidth if the CBR arrival rate approaches the link bandwidth. Fur-

---

<sup>1</sup> A flow is defined by IP-address pair and port numbers, respectively flow-id. Note that a flow’s friendliness is a relative notion, dependent on the conservatism of the flow’s congestion control mechanism compared to its competitors for the network resources.

<sup>2</sup> A CBR flow transmits with a constant bit rate into the net. CBR is not related to the ATM service.

\* This work is partly sponsored by TA and FFF Austria

thermore, unresponsive flows cause congestion collapse in scenarios where a congested link, transmitting only packets of unresponsive flows, is followed by a lower bandwidth successor link. In these scenarios packets of unresponsive flows, having consumed the total bandwidth at the first link and thereby shut out friendly flows, are dropped at the router output port served by the second lower-bandwidth link. As a consequence, the overall throughput is limited by the capacity of the low-bandwidth link. More information about kinds of unfriendly flows and congestion collapse can be found in [2].

We are aware of three different approaches to solve the problem of unfriendly flows:

- *Resource pricing* [3] enforces the user to pay a small charge for each congestion indication received from the net. Unfriendly flows have higher transmission rates in times of congestion than friendly flows hence they receive more congestion indications, causing higher accounts. Having to pay fines for unfair resource usage is clearly a strong incentive for users to behave well.
- The *allocation approach* isolates flows from each other by ensuring a fair allocation of resources to each flow. Thus an unfriendly flow, trying to consume more than the fair share merely increases its own packet-loss rate but does not harm other flows. An example for the allocation approach, distributing link capacity according to the max-min fairness criterion [6], would be a network of routers having per flow fair-queueing schedulers [4] or one of its derivatives (e.g. [5]) implemented. Note that merely restricting flows to their fair share does not give users a strong incentive to behave well and to generate friendly flows but avoids unfairness in case they are misbehaving. Additionally, as shown by simulation in [2], the problem of bandwidth wastage due to unresponsive flows persists in a max-min fair allocated network having a higher bandwidth link with a lower bandwidth successor link utilized by many unresponsive flows. Due to their large number, the unresponsive flows grab the major portion of the bandwidth at the first, high-bandwidth link and experience vast packet losses at the second, lower-bandwidth link.
- As opposed to allocation, the *identification approach* attempts to explicitly identify unfriendly flows by some remarkable behavior. Such a behavior could for instance be an extraordinary high demand compared to other flows in times of congestion or an improper reaction to packet loss relative to a flow behavior considered as friendly [2]. Flows identified as unfriendly can be restricted to their fair share, to some small portion of the bottleneck-link capacity or, in the extreme case, are completely shut out at congested links. The stronger the discrimination, the stronger the incentive for users to behave well. Note that the identification approach is inherently more lightweight than the allocation approach as it does not necessitate knowledge of the instantaneous load situation in the net for the computation of a fair resource allocation. As obtaining this knowledge is left to the user, identification mechanisms implemented in routers may determine the degree of friendliness of a flow in low-priority background tasks having rather long time scales [2]. On the other hand, identification requires a roughly homogeneous behavior of end-to-end congestion control mechanisms in order to provide fairness as the network only identifies and discriminates unfriendly flows, but does not explicitly allocate resources to friendly flows. For the Internet with its huge base of installed TCP agents this implies that congestion control mechanisms have to be TCP-friendly, at least in the short term.

The paper is organized as follows: section 2 summarizes related work, section 3

explains and discusses the operational principles of the MUV algorithm and outlines possible modifications. In section 4 MUV's ability to identify and discriminate unfriendly flows is investigated by simulation and section 5 concludes this paper.

## 2 Related Work

The Random Early Detection (RED) algorithm [7] employs the parameter set  $\{minth, maxth, maxp\}$  in order to probabilistically drop packets arriving at a router output-port. If the average queue-size ( $avg$ ) is smaller than  $minth$  no packet is dropped. If  $minth < avg < maxth$ , RED's packet-drop-probability varies between zero and  $maxp$ . If  $avg > maxth$ , each arriving packet is dropped. WRED [8] and RIO [9], both enhancements of RED intended for service differentiation in the Internet [10], relate arriving packets to the parameter set  $\{minth_{in}, maxth_{in}, maxp_{in}\}$ , respectively  $\{minth_{out}, maxth_{out}, maxp_{out}\}$  if the packet has been marked as "in-profile", respectively "out-of-profile" according to its flow's service profile at a network boundary. Marking a packet means writing a well-defined number (a diff-serv codepoint) into its IP-header's Type-of-Service field. Assuming  $minth_{in} \geq maxth_{out}$  in-profile packets are accommodated with a high probability while all out-of-profile packets are dropped if  $avg > maxth_{out}$ . Hence out-of-profile packets are discriminated against in-profile packets. As opposed to WRED, which uses one average queue size for all packets in the queue, RIO computes an extra average queue-size only for in-profile packets.

Numerous authors [14][15][16][17] have investigated a simple stationary model of TCP congestion avoidance [12][13] deriving the rate of a TCP-flow ( $R_{TCP}$ ) as a function of the packet-drop probability ( $p$ ), the maximum segment size ( $MSS$ ) and the round-trip time ( $RTT$ ):

$$R_{TCP} = \frac{c \cdot MSS}{RTT \cdot \sqrt{p}} \quad (1)$$

The constant  $c$  "varies" between 0.93 and 1.31 depending on the loss model and whether the delayed-ACK option [18] is turned on or off. The number of packets a TCP data sender has to transmit until equilibrium is achieved and equation (1) becomes valid equals  $1/p \cdot \ln(1/c \cdot \sqrt{p})$  [17]. However, the model does not consider the influence of TCP retransmission timeouts followed by Slowstart [12] hence (1) only holds as an upper bound for TCP performance in realistic environments [17]. A more rigorous model, taking retransmission timeouts into account has been presented in [19].

In [2] routers execute a low-priority background task in periodic time intervals. Among others, the "TCP-friendly test" is applied to incoming traffic in order to identify unfriendly flows. A flow is considered as non TCP-friendly if its arrival rate is greater than its equivalent TCP-friendly rate, i.e. the arrival rate of a TCP flow having the same  $RTT$ ,  $MSS$  and  $p$ . A flow's arrival rate ( $A$ ) is estimated from the RED packet-drop history [20]. The TCP-friendly rate is calculated via equation (1) by the background task in periodic intervals of time. As routers generally do not have knowledge of a flow's end-to-end round-trip-time the TCP-friendly test in [2] sets the  $RTT$  value for computation of the TCP-friendly rate at twice the outgoing link's delay, which may heavily underestimate the end-to-end RTT. Therefore the TCP-friendly rate is overestimated and unfriendly flows are unlikely to be detected in scenarios where the outgoing link's delay is not the dominant portion of the end-to-end delay [5].

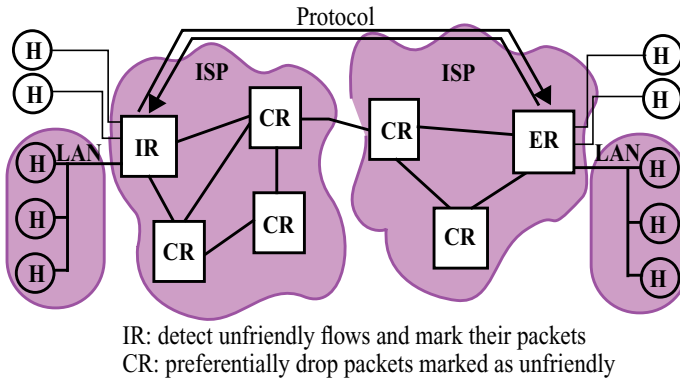
Other approaches for identification and discrimination of unfriendly flows, requiring restricted amount of per flow state and a flow lookup in core routers have been proposed in [5][21][22] and [23].

### 3 MUV Algorithm

#### 3.1 Network Model and operational Principle of MUV

We define the *ingress router* (IR) as the first router of an ISP traversed by packets of a unidirectional flow on their way to the destination. In analogy, the *egress router* (ER) is defined as the last router. IR and ER can be considered as *edge routers*, located at the boundary of ISPs to a private domain. All other routers along a flow's path between IR and ER are called core routers (CR). Hosts (H) are attached to IR and ER by subscriber lines or private LANs.

MUV identifies unfriendly flows by comparing per-flow arrival rates with TCP-friendly rates as proposed in [2] and outlined in section 2. However, contrary to [2], the TCP-friendly test is only performed at the IR, measuring the flow's packet-drop probability ( $p$ ) and round-trip-time ( $RTT$ ) between IR and ER via a light-weight protocol with the ER. The measurements of  $p$  and  $RTT$  are used as approximations of the flow's end-to-end round-trip-time and drop probability for computation of the TCP-friendly rate. If a flow's arrival rate is greater than its TCP-friendly rate the flow is considered as unfriendly and its packets are marked at the IR<sup>3</sup>. Core routers discriminate packets marked as unfriendly with drop-preference algorithms like WRED or RIO. The principle of MUV is illustrated in figure 1:



**Fig. 1.** Network model and operation of MUV

The operational principle of MUV shows its compliance to the Differentiated Services Architecture [10] of the Internet. Per flow state is only required at the edge of the network; core routers do not have to store per-flow information or perform a flow lookup.

The level of punishment flows detected as unfriendly should experience can be determined by adjusting the parameters of the drop-preference algorithms in the core routers. Our position is that unfriendly flows should be completely shut out in order to give users

<sup>3</sup> Packet marking can be done as explained in section 2 for RIO or WRED.

a strong incentive to behave well and to avoid congestion collapse due to unresponsive flows getting some residual throughput on congested links (see section 1).

### 3.2 MUV Algorithm

The MUV algorithm assumes a background task at IR and ER that periodically scans a hash table with stored per-flow information, changes per-flow states and sends signalling messages. The period length of the background task (i.e. the time interval between subsequent passes of the background task at a given index of the hash table where per-flow information is stored) is assumed to be in the range of several round-trip-times of flows. Figure 2 and the subsequent paragraphs explain the MUV algorithm by examining its sample operation for an unfriendly high-demand flow on a congested network. Although this “explanation by example” is admittedly incomplete we believe that it facilitates the understanding of the algorithm. The vertical axis in figure 2 represents time. The short evenly spaced horizontal lines crossing the vertical time lines indicate the background task performing some actions on the flow.

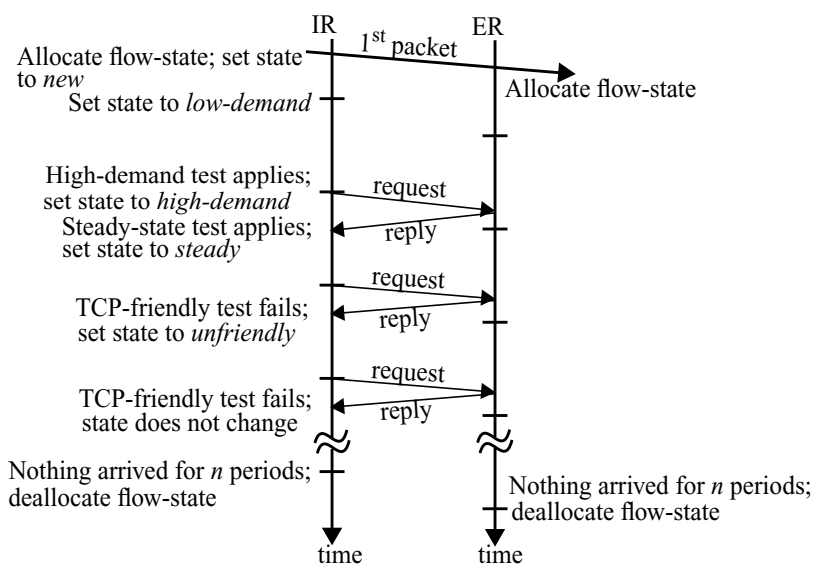


Fig. 2. MUV Algorithm

When the flow transmits its first IP packet, per-flow information (IP-Addresses, Port-numbers and several quantities required by MUV) is allocated in hash tables at IR and ER. Starting with the flow's first packet, the IR measures the flow's demand ( $D$ ) and the ER measures the flow's throughput ( $T$ ) simply by incrementing a counter at each packet arrival by the packet size. At the IR the flow is set to state *new*. At the subsequent pass of the background task the flow is set to state *low-demand*. When the background task passes a flow in state *low-demand* (this happens one period later) the “high-demand test” is performed. The high-demand test compares the flow's arrival rate ( $A$ ) with the “high-demand threshold”. The arrival rate is computed as the flow's demand in bytes divided by the current time minus the time the flow's first packet has been received at the IR. As our flow is assumed to have high demand its arrival rate is greater than the high-demand threshold, hence the flow's state is changed to *high-demand*.

For flows in state *high-demand* the MUV Protocol is executed between IR and ER. Each period, the IR sends a request message including a sequence number to the ER<sup>4</sup> and the ER answers with a reply message. Request and reply messages are short ICMP packets, filtered out by the edge routers after their evaluation. The reply message includes the flow's throughput  $T$  since its first packet arrival at the ER and the sequence number of the request message. When the IR receives the reply message it checks if the sequence number of the last-sent request message and the reply message are identical. This happens to be true in our example, hence the IR may measure the flow's packet-drop-probability  $p$  as follows:  $p = (D - T) / D$ . Now the IR may determine whether the flow is in steady state. As mentioned in section 2, a TCP flow is in steady state when  $1/p \lg(1/c\sqrt{p})$  packets have been sent. The IR then checks if  $D > 1/p \lg(1/c\sqrt{p})$  and changes the flow's state to *steady* as we assume this condition to be fulfilled. The instantaneous values of  $D, T$  and the current time are stored in order to allow computation of  $A^s$  and  $p^s$ , denoting arrival rate and drop probability since the point in time the flow is assumed to be in steady state.

Now, that the MUV agent at the IR knows that this flow would be in steady state if it was a TCP flow, the "TCP-friendly test" may be performed by comparing the flow's steady state arrival rate with its equivalent TCP-friendly rate. On the receipt of the next reply message the IR updates  $A^s$  and  $p^s$  and computes the TCP-friendly rate  $R_f$ , where  $R_f = c \text{MSS} / (RTT_{IR-ER} \sqrt{p^s})$  (see equation (1), section 2). As we assume our flow to be unfriendly, MUV detects that  $A^s \geq \text{nice } R_f$  and sets the flow's state to *unfriendly*. The constant parameter  $\text{nice} \geq 1$ , determines MUV's tolerance against non TCP-friendly flows. If  $\text{nice}$  equals one in the most intolerant case, flows having an arrival rate higher than the TCP-friendly rate will be determined as unfriendly. The flow's end-to-end RTT is approximated by the mean RTT between IR and ER ( $RTT_{IR-ER}$ ) which can be measured by subtracting the departure time of the request message from the arrival time of the corresponding reply message.

As our flow has been detected as non TCP-friendly, the IR marks its packets from now on. Core routers have a drop-preference mechanism like WRED or RIO implemented. For best-effort packets marked as unfriendly the parameter set  $\{\text{minth}_{uf}, \text{maxth}_{uf}, \text{maxp}_{uf}\}$ , for all other best-effort packets the parameter set  $\{\text{minth}_f, \text{maxth}_f, \text{maxp}_f\}$  is valid. The network is assumed to be congested therefore the average queue size converges between  $\text{minth}_f$  and  $\text{maxth}_f$  at congested routers along the flow's path. As  $\text{minth}_f$  is assumed to be greater than  $\text{maxth}_{uf}$  the packet-drop-probability for our flow's packets equals one. In other words, the flow is completely shut out.

On the arrival of the next and all subsequent reply messages the IR again updates  $A^s$ ,  $p^s$ ,  $RTT_{IR-ER}$  and performs the TCP-friendly test in order to reset the flow's state to *steady* in case the flow has been falsely detected as unfriendly due to some short-term bursty behavior. However, our flow is unfriendly by assumption hence it does not change state and its packets are continued to be dropped at congested core routers until the flow terminates. In order to avoid keeping per-flow information forever when the flow has terminated, the background task at IR and ER deallocates the flow's state in case no packet has arrived for more than  $n$  periods, where  $n$  is a constant parameter.

---

<sup>4</sup> Strictly speaking, the IR sends the request message to the destination of the flow as it does not know the IP Address of the ER.

### 3.3 Discussion of the MUV Algorithm

[24] provides a detailed analysis of MUV properties like time scales of control, MUV and ECN, why preferential packet dropping and not preferential packet scheduling has to be used in core routers, how MUV can be faked, deployment issues and how flow state is deallocated in edge routers. Additionally, modifications of MUV minimizing signalling overhead between edge routers are discussed. Due to its limited size, however, we are only able to give an overview on MUV properties and modifications in this paper.

We illustrate the significance of the state *new* by assuming the high-demand test was already performed at the first time the background task passes a flow. As the allocation of flow state in the hash table appears to be asynchronously to the execution of the background task it might happen that the high-demand test is performed on a flow which was allocated just some milliseconds ago. As a consequence, the high-demand test would be performed on the basis of a measurement over an interval of time too short to be relevant. Setting flows into state *new* at the storage of flow state and thereby waiting at least for one period length until the high-demand test is executed avoids considering low-demand flows having only a short burst at their start as high-demand.

The task of the high-demand test is to distinguish among high- and low-demand flows and thereby minimize the signalling overhead caused by the request/reply messages. The majority of Internet flows are short and have low demand. Hence it is desirable to police only those flows which may potentially harm the network and to avoid unnecessary signalling for flows having low demand or only a few packets to send. Setting the high-demand threshold is a trade-off between high signalling overhead and the risk of congestion collapse due to unresponsive flows which persists as long as unresponsive best-effort flows may send into the network.

The friendliness of a flow's congestion control algorithm does not change over time compared to others competing for the bandwidth. From this simple observation follows that the quantities needed for the computation of the TCP-friendly rate ( $A^s, p^s, RTT_{IR-ER}$ ) may be averaged over the maximum possible time interval - the total time a flow has been in steady state. As MUV merely identifies unfriendly flows, there is no need to be aware of the instantaneous load situation in the net. Hence MUV may operate in time scales of several seconds rather than one RTT like congestion control mechanisms. Due to its long time scales the MUV algorithm is stable against oscillations and may perform measurements with a rather low frequency, keeping signalling overhead low. A second consequence of our observation is that mean averaging over the total time a flow has been in steady state should be used for computation of  $A^s, p^s$  and  $RTT_{IR-ER}$  and not averages giving recent occurrences more weight (e.g. exponentially weighted moving averages).

The MUV algorithm is based on two assumptions in order to work properly. First, the packet-loss probability between hosts and edge routers is assumed negligible compared to the packet drop-probability between IR and ER. Although this assumption is likely to hold as the WAN and not the LAN is the bottleneck in most scenarios, it may fail. IR and ER have no means to detect these packet drops. Second, the delay between hosts and edge routers is assumed negligible compared to the delay between the edge routers. In case the delay between IR and ER is not the dominant part of the end-to-end delay, the measured RTT between IR and ER significantly underestimates the end-to-end RTT. If the end-to-end packet-drop rate and/or RTT is underestimated, the TCP-friendly rate ( $R_f$ ) overestimates the rate of a TCP flow ( $R_{TCP}$ ) as predicted by equation (1) and unfriendly flows could eventually not be detected. Underestimation of the end-to-end RTT can be



avoided by a simple enhancement. Operators of ingress- and egress routers are assumed to have a rough idea of the average delay to attached hosts and local area networks. Hence the RTT between source and IR ( $RTT_{source-IR}$ ) and the RTT between ER and sink ( $RTT_{ER-sink}$ ) could be stored as constant parameters at IR and ER. The ER could communicate  $RTT_{ER-sink}$  to the IR in reply messages and the IR could estimate the end-to-end RTT as the sum of  $RTT_{source-IR}$ ,  $RTT_{IR-ER}$  and  $RTT_{ER-sink}$ .

The choice of the period length means balancing a trade-off between slower detection of unfriendly flows and less accurate measurements of the quantities required for computation of the TCP-friendly rate and more signalling overhead due to a higher measurement frequency. [2] recommends a lower bound of several round-trip-times for the period length in order to guarantee sufficiently long time intervals between measurements. Like [2], we set the period length to 5 seconds.

As shown in [17] the upper bound provided by equation (1) is rather reliable but the TCP rate is overestimated in case of frequent retransmission timeouts. For the TCP-friendliness test, these properties of the model imply that TCP flows are unlikely to be falsely detected as unfriendly. However, unfriendly but responsive flows, being somewhat more aggressive than TCP, may not be detected in case of severe congestion and frequent retransmission timeouts.

MUV is aware of approximations of a flow's RTT, throughput and drop-probability. Thus arbitrary criteria could be used for identification of unfriendly flows in addition to the TCP-friendliness criterion. For instance, a flow could be considered as unfriendly in case it does not pass the TCP-friendliness *or* the unresponsiveness test proposed in [2].

The MUV algorithm as proposed above requires two ICMP messages for signalling per period and per flow having an arrival rate greater than the high-demand threshold. There are possibilities to improve the scalability of the algorithm and to make it work with flow aggregates (typically the traffic coming from a LAN or a subscriber-line user). For instance, a predefined number of highest bandwidth flows could be filtered out of a flow aggregate at the IR and MUV could be performed only for these filtered flows. If flows are identified as unfriendly the total aggregate, or only the identified flows could be discriminated. To further reduce signalling overhead a flow could be defined by IP-address pair or network-address pair instead of IP-address pair and port numbers. Additionally, there is no explicit need to use packets for the signalling of request messages. This could be done "in-band" using header fields or a new kind of IPv6 extension header.

## 4 Simulations

We have implemented the MUV algorithm into the ns simulator [26]. Simulations in [24] investigate MUV's behavior in scenarios with multiple congested gateways, varying number of flows, varying link capacities and delays, various traffic patterns aggregating CBR, ON/OFF flows, aggressive TCP flows, TCP-Reno and SACK-TCP flows with and without ECN. Queue-size over time graphs show why packets marked as unfriendly are shut out at congested output ports of core routers by WRED. Due to space limitations, however, it is only possible to integrate a subset of these simulations into this paper.

Simulations are performed on a network topology as illustrated in figure 3. The following traffic pattern is used for all simulations in this paper. Host H1 sends background traffic consisting of 10 TCP flows (flow-IDs 4-13) to host H2. The TCP flows start between zero and 30 seconds of simulation time and last for the entire simulation. Host

H3 generates cross traffic in forward direction consisting of 10 TCP flows (flow-IDs 100-109) with varying life-times to host H4. Host H5 sends 10 reverse traffic TCP flows (again with varying lifetimes) to H6. Background traffic, reverse- and cross-traffic TCP flows are a mix of TCP-Reno and SACK flows and have different packet sizes and send-window sizes [18]. The demand of reverse- and cross-traffic has a maximum between 30 and 55 seconds (causing congestion at the link CR3 to CR2) and approaches zero until approximately 80 seconds of simulation time. Permanent queueing happens at the congested output port of CR1 served by the bottleneck link in direction CR2.

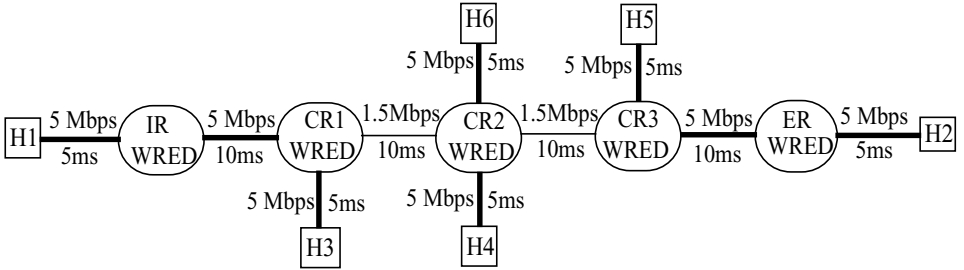


Fig. 3. Simulated network

As shown in [27], RED's aggressiveness regarding packet dropping has to be adjusted dynamically to the number of flows traversing the output port of a congested router in order to avoid convergence of the average queue-size to the minimum or maximum queue-size threshold. RED's aggressiveness can be adjusted by increasing the  $max_p$  parameter if  $avg$  becomes smaller than  $min_{th}$  and decreasing  $max_p$  in case  $avg$  becomes greater than  $max_{th}$ . For simulations in this paper a modified version of WRED with adoption of  $max_p_f$  has been implemented. Similar to [27], we have set the decrease/increase factor for  $max_p_f$  to 2.

Other WRED parameters are set as follows: *byte-mode* is activated,  $w_q = 0.002$ ,  $min_{th}_{uf} = 5000$  bytes,  $max_{th}_{uf} = 9000$  bytes,  $max_{p_{uf}} = 1$ ,  $min_{th}_f = 10000$  bytes,  $max_{th}_f = 17500$  bytes,  $max_{p_f} = 1/50$ ,  $mean-pktsz = 500$  bytes,  $total\ buffer\ size = 25000$  bytes. We refer to [7] for an explanation of the RED parameters  $w_q$  and  $mean-pktsz$ .

In order to be able to evaluate MUV's behavior in the presence of non TCP-friendly but responsive flows, we have modified the simulator's TCP-Reno code to a version of TCP we call "TCP-nasty". On the contrary to TCP-Reno [13], TCP-nasty does not halve the congestion window in response to one packet drop per window but multiplies the window by a constant  $\alpha$ , where  $0.5 \leq \alpha < 1$ . Additionally, TCP-nasty does not increase the congestion window by one MSS per RTT in case no drop happens (as TCP-Reno does), but by  $\beta$  MSS (where  $\beta \geq 1$ ). In analogy to [17] and equation (1) in section 2, we have derived a stationary performance model for the rate of a TCP-nasty flow ( $R_{nasty}$ ), substituting the window-decrease of 0.5 with  $\alpha$ , and the window-increase with  $\beta$  MSS. Let the *unfriendliness-factor* ( $\phi$ ) of a TCP-nasty flow be defined as  $\phi = R_{nasty}/R_{TCP}$ . Hence an unfriendliness-factor of  $n$  means that a TCP-nasty flow has a rate of  $n$  times the rate of a conforming TCP-Reno flow, as predicted by the model. It has been shown in [24] that  $\phi$  is related to  $\alpha$  and  $\beta$  as follows:

$$\phi = \frac{\sqrt{2\beta(1-\alpha^2)}}{(1-\alpha)2c} \quad (3)$$

We have implemented the modification of computing the RTT estimation as  $RTT_{source-IR} + RTT_{IR-ER} + RTT_{ER-sink}$  (see section 3.3) in the ns simulator. MUV parameters for all simulations, except otherwise noted, are set as follows:  $nice = 1.5$ ,  $high-demand-threshold = 2000$  bytes,  $RTT_{source-IR} = 4$  ms,  $RTT_{ER-sink} = 4$  ms, period length 5s.

#### 4.1 MUV with CBR and TCP-nasty Flows

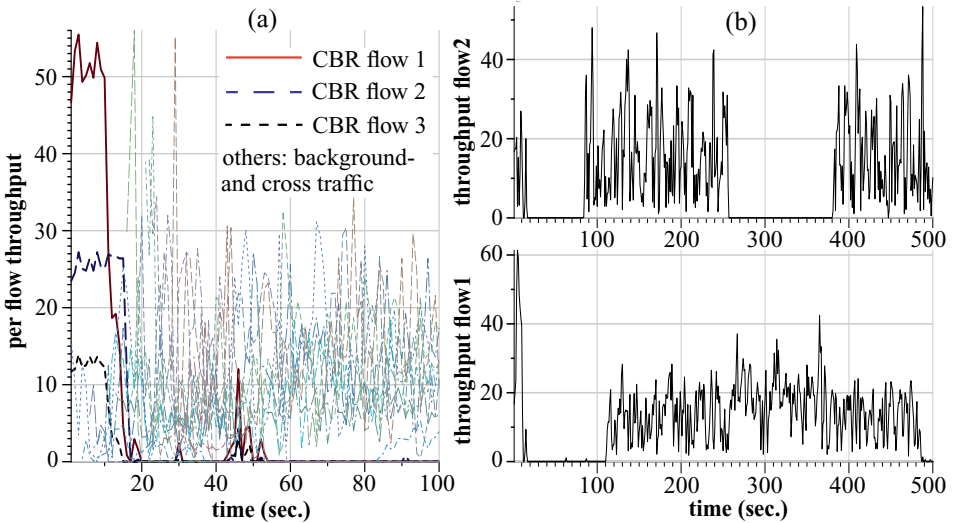


Fig. 4. Per flow throughput at the bottleneck link as a percentage of bottleneck capacity

Figure 4a shows a simulation with MUV and unresponsive CBR flows. CBR flows 1,2,3 are sent from H1 to H2, start at 0 seconds of simulation time, never terminate, have a packet size of 500 bytes and send at rates of 800,400,200 kbps, respectively. According to their arrival rates, the CBR flows grab the major part of the bottleneck capacity during the first seconds. At approximately 11 seconds of simulation time flows 1 and 3 are identified as non-TCP friendly, their packets are marked as unfriendly and they are shut out at the congested router (CR1). Flow 2 is identified one period later as one of MUV's signaling packets for flow 2 gets lost. Background, cross- and reverse TCP flows frequently perform Slowstart or quit sending due to their short lifetime between 45 and 55 seconds of simulation time, causing considerable oscillations of the queue size at the bottleneck. As a consequence the average queue size is frequently decreased below  $maxth_{uf}$  and the CBR flows, although still in state unfriendly, receive marginal throughput.

Figure 4b shows a simulation with MUV and responsive but unfriendly flows. TCP-nasty flows 1,2 start sending at 0 seconds of simulation time, have packet sizes of 500 bytes, send windows of 100 packets and an unfriendliness factor of 3 and 6, respectively. Flow 2 is detected as unfriendly at approximately 11 seconds of simulation time. From now on, all its packets are discarded. As a consequence, the TCP-nasty source decreases its congestion window and thereby its transmission rate into the net. At approximately 80 seconds, MUV detects that the average arrival rate of flow2 is below its TCP-friendly rate and sets the flow into state *steady*. Hence flow2's packets are no longer dropped at the core router, its congestion window is increased again and the flow receives a portion of

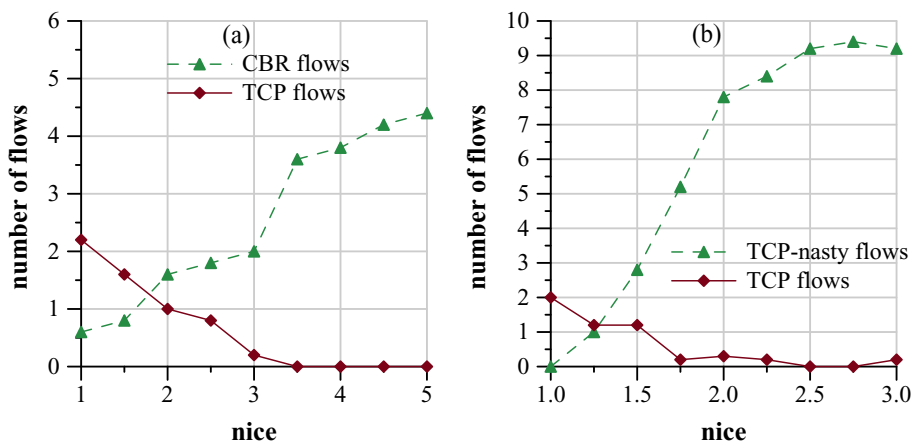
the link capacity according to its unfriendliness-factor (between 85 and 255 seconds of simulation time). When flow2's average arrival rate exceeds its TCP-friendly rate again (this happens at 255 seconds), the flow is set to state *unfriendly* again and its packets are marked and dropped by WRED at the bottleneck router. This shows that MUV periodically shuts out and subsequently accommodates unfriendly but responsive flows. The longer the flow exists, the longer the period lengths, as MUV computes its quantities for the TCP-friendly test over a longer time interval.

Comparing flows 1 and 2 in figure 4b shows that the length of the accommodation period is inversely proportional to the unfriendliness factor of a flow. If a flow with higher  $\phi$  is reset into state *steady* again, its accommodation period is shorter as the flow increases its transmission rate more aggressively. For instance, in figure 6 the first accommodation period of flow 2 lasts from 85 to 255 seconds whereas the first accommodation period of flow 1 lasts from 110 to 485 seconds of simulation time.

## 4.2 Unfriendly Flows and TCP Flows identified as a Function of nice Parameter

Additional traffic sent from host H1 to host H2:

- figure 5a: 10 CBR flows 1,2,...,10 with rates of 40, 80, 120, ..., 400 kbps.
- figure 5b: 10 TCP-nasty flows 1,2,...,10 with unfriendliness-factor  $\phi = 1.5, 2, 2.5, \dots, 6$ .



**Fig. 5.** Number of conforming TCP flows detected, number of unfriendly flows not detected as a function of the nice parameter

Figure 5 can be interpreted as follows:  $n$  unfriendly flows (CBR in figure 5a, TCP-nasty in figure 5b) not detected means flows 1,2,..., $n$  have not been detected (as they have the lowest arrival rates, respectively unfriendliness factors) and flows  $n+1, n+2, \dots, 10$  have been detected as unfriendly. Values are computed over the total duration of the simulation (100 seconds);  $n$  is computed as a mean over 5 simulations. No flow terminates prior to the simulation. All flows start at 0 seconds of simulation time

In addition to the unfriendly flows, background traffic consisting of 10 TCP flows traverses the IR and the ER. These flows might be falsely identified as unfriendly as the TCP model (see section 2, equation (1)) used for the computation of the TCP-friendly rate may fail in realistic scenarios. In case MUV sets a TCP flow to state *unfriendly*, the flow experiences high packet loss, backs off and is reset to state *steady* a few periods

later. Dimensioning the *nice* parameter means balancing a trade-off between falsely identifying conforming TCP flows as unfriendly and not detecting unfriendly flows. As *nice* is increased fewer conforming TCP flows are falsely identified as unfriendly, but at the same time fewer unfriendly flows are detected.

Figure 5a shows that MUV reliably identifies unresponsive CBR flows even with moderate arrival rates close to the fair share. For instance, if *nice* equals 3 the average number of CBR flows not detected equals 2, i.e. on the average only the 40 kbps and the 80kbps CBR flows are not identified as unfriendly.

Figure 5b: although less conservatively than TCP-Reno, TCP-nasty flows back off in case of congestion hence their identification by MUV exhibits higher sensitivity to the setting of *nice* than the identification of CBR flows. With *nice* equal two, 7.8 TCP-nasty flows (i.e. flows with  $\phi < 5$ ) are not detected on the average. This shows that we can only expect to identify flows having significantly less conservative congestion-control algorithms implemented than TCP.

Regarding false identification of conforming TCP flows, MUV's behavior is similar in figure 5a and figure 5b. We conclude from figure 5 that setting *nice* between 1.5 and 2 should minimize the caveats of false identification of TCP-friendly and non-identification of unfriendly flows.

## 5 Conclusions

As shown in this paper, it is feasible to implement mechanisms in the Internet that are able to identify and discriminate non TCP-friendly flows in most scenarios. Hence, identification of non TCP-friendly flows can be considered as an interesting alternative to its competitor approaches resource allocation and resource pricing in order to give Internet users an incentive to be well-behaving instead of misbehaving.

We have outlined how MUV identifies and discriminates non TCP-friendly flows. Simulations indicate that MUV reliably detects and punishes unresponsive flows even if their arrival rates are only marginally higher than the fair share. Hence MUV solves the problems of unfairness and congestion collapse due to unresponsive best-effort flows. Additionally, MUV identifies and restricts flows using congestion control algorithms significantly less conservative than TCP congestion control.

The costs for these features seem to be acceptable. MUV only stores per-flow state at the edge of the network. It can therefore be considered as scalable regarding per-flow state and complexity in the data-forwarding path of core routers. Due to its design, MUV fits into the Differentiated Services Architecture of the Internet. The non-optimized version of MUV proposed in this paper requires two messages for signalling per high-demand flow every 5 seconds. As outlined in section 3.3, simple modifications to the MUV algorithm should significantly reduce this signalling overhead without significant loss of performance.

## References

1. B. Braden, D. Clark et al., "Recommendations on Queue Management and Congestion Avoidance in the Internet", RFC 2309, April 1998
2. S. Floyd, K. Fall, "Promoting the Use of End-to-End Congestion Control in the Internet", Submitted to IEEE/ACM Transactions on Networking, February 1998

3. R.J. Gibbens, F.P. Kelly, "Resource Pricing and the Evolution of Congestion Control", <http://www.statslab.cam.ac.uk/~frank/evol.html>
4. A. Demers, S. Keshav, S. Shenker, "Analysis and Simulation of a Fair Queueing Algorithm", Proceedings of ACM SIGCOMM, 1989
5. I. Stoica, S. Shenker, H. Zhang, "Core-stateless Fair Queueing: achieving approximately fair Bandwidth-Allocations in High-Speed Networks", Proceedings of ACM SigComm, 1998
6. K. K. Ramakrishnan, D. Chiu, R. Jain, "Congestion Avoidance in Computer Networks with a connectionless Network Layer; Part 4, A selective binary feedback scheme for general topologies", DEC-TR-510, 1987
7. S. Floyd, V. Jacobson, "Random Early Detection Gateways for Congestion Avoidance", IEEE/ACM Transaction on Networking, August 1993
8. Cisco Web-pages, [http://www.cisco.com/warp/public/732/netflow/qos\\_ds.html](http://www.cisco.com/warp/public/732/netflow/qos_ds.html)
9. D. Clark, "Explicit Allocation of Best Effort Packet Delivery Service", <http://www.ietf.org/html.charters/diffserv-charter.html>
10. S. Blake et al., "An Architecture for Differentiated Services", RFC 2475, December 1998
11. K. Nichols et al., "Definition of the Differentiated Services Field (DS Field) in the IPv4 and IPv6 Headers", Internet Draft, August 1998
12. V. Jacobson, "Congestion Avoidance and Control", Proceedings of ACM SIGCOMM Conference, August 1988
13. V. Jacobson, "Modified TCP Congestion Avoidance Algorithm", Message to end2end-interest mailing list, April 1990
14. S. Floyd, "Connections with multiple congested Gateways in Packet-Switched Networks", Part one: One-way Traffic", Computer Communications Review, Oct. 1991
15. T.V. Lakshman, U. Madhow, "The Performance of TCP/IP for Networks with High Bandwidth-Delay Products and Random Loss", IFIP Transactions, High Performance Networking, 1994
16. T. Ott, J.H.B. Kempermann, M. Mathis, "The stationary Behavior of ideal TCP Congestion Avoidance", Preprint, August 1996
17. M. Mathis, J. Semke, J. Mahdavi, "The Macroscopic Behavior of the TCP Congestion Avoidance Algorithm", Computer Communications Review, July 1997
18. W. R. Stevens, "TCP-IP Illustrated, Volume 1", Addison Wesley, 1994
19. J. Padhye et al., "Modeling TCP Throughput: A simple Model and its empirical Validation", Proceedings of ACM SIGCOMM, August 1998
20. S. Floyd, K. Fall, K. Tieu, "Estimating Arrival Rates from the RED Packet Drop History", Draft, March 1998
21. D. Lin, R. Morris, "Dynamics of Random Early Detection", Proceedings of ACM SIGCOMM, 1997
22. T. Ziegler, S. Fdida, U. Hofmann, "RED+ Gateways for Identification and Discrimination of unfriendly best-effort Flows in the Internet", Proceedings of IFIP Broadband Communications 99, November 1999
23. T.J. Ott, T.V. Lakshman, L.H. Wong, "SRED: Stabilized RED", Proceedings of IEEE Infocom, 1999
24. S. Deering, R. Hinden, "Internet Protocol, Version 6 (IPv6) Specification", RFC 2460, December 1998
25. T. Ziegler, S. Fdida, "A distributed Mechanism for Identification and Discrimination of non TCP-friendly Flows in the Internet", extended version of this paper, unpublished, November 1999, <http://www-rp.lip6.fr/~costa/production.html>
26. NS Simulator Homepage, <http://www-mash.cs.berkeley.edu/ns/>
27. W. Feng et al., "Techniques for eliminating Packet Loss in congested TCP/IP Networks", University of Michigan, CSE-TR-349-97, November 1997

# New Handoff Strategies in Microcell/Macrocell Overlaying Systems

Selma Boumerdassi, André-Luc Beylot

Laboratoire PRiSM, Université de Versailles - Saint-Quentin  
45, avenue des Etats-Unis, 78035 Versailles Cedex France  
E-mail: Selma.Boumerdassi, Andre-Luc.Beylot@prism.uvsq.fr

**Abstract.** To cope with the increasing demand of mobile and personal communications, two-tier cellular systems are likely to be deployed in major city center. Reservation schemes are consequently to be adapted to such a context: forced call terminations due to handoff call blocking are generally more objectionable than new call blocking. In order to maintain an acceptable call dropping probability rate, we propose, below, a new adaptive guard channel scheme : Hierarchical Predictive Reservation Policy (HPRP). In this scheme, the number of reserved channels depends on the actual number of calls in progress in the neighboring cells. In micro/macrocels systems, users are assigned to a given level of the hierarchy according to their speed. In this paper, three schemes are proposed and compared by discrete event simulations. The efficiency of the proposed methods is emphasized on a complex configuration.

## 1 Introduction

Personal Communication Networks (PCN) have emerged as an important field of activity in telecommunications [9]. Among all the encountered problems, the implementation of broadband features on the wireless medium is one of the most difficult to handle: a large bandwidth is required to offer such services with a quality of service similar to those of wired networks. Future PCNs will employ micro-cells and pico-cells to support a higher capacity, and thus increasing the frequency of handoff calls and dramatizing the effects of non-uniform traffic conditions due to mobility patterns in the covered regions (roads, dead-ends, ...). Hierarchical microcell/macrocell architectures have been proposed for future personal communication systems. These architectures provide capacity relief to a macrocell system and offer many advantages over a pure microcell system [11]. Furthermore, the fast handoff requirement in a pure microcell system can be relieved in the overlaying architecture by temporarily connecting the call to a macrocell Base Station (BS). In a two-tier system (of microcells overlaid with umbrella macrocells), low mobility users (e.g. pedestrians) should be assigned to microcells and undergo handoffs when crossing microcell boundaries, while high mobility users (e.g. cars) should be assigned to macrocells and undergo handoffs when crossing macrocells boundaries. In this paper, different admission strategies are considered.

To support network-wide handoffs, new and handoff calls will compete for requesting resources. Handoff calls require a higher congestion-related performance, i.e. blocking probability, with regard to new calls in order to minimize subscriber dissatisfaction. Different methods for accessing resources were proposed [14] [1]. In mobile networks, one common bandwidth resource access priority scheme is the guard channel scheme [5] [3] [4] [12]. In this paper, we propose a new dynamic guard channel scheme: Hierarchical Predictive Reservation Policy (HPRP). In this scheme, the number of guarded channels depends on the actual traffic parameters in order to achieve the Quality of Service requirements for both new call blocking and handoff blocking probabilities. A multi-period traffic representation has been considered. The values of the HPRP parameters have to be optimized for each time period. In HPRP, the number of reserved channels depends on the number of ongoing calls in neighboring cells (i.e. micro/macro cells) and on the routing probabilities of mobiles.

Discrete event simulations of these schemes were run. We demonstrate the effectiveness of our reservation method thanks to a complex and realistic multi-cell configuration, and we present performance evaluation under different traffic loads.

This paper is organized as follows. Section 2 describes the adaptive guard schemes. In section 3, the simulation model is presented and the performance results of the proposed guard algorithm and of the different admission strategies are analyzed. We conclude the paper in section 4.

## 2 Models for adaptive guard bandwidth schemes and admission strategies

### 2.1 Introduction

The fixed guard bandwidth scheme is a prioritized resource access scheme which allows new calls and handoff calls to share the capacity, by giving a resource access priority to handoff calls. Since a larger resource capacity limit is given to handoff calls, the guard channel is defined as the difference between the two capacity limits. In this paper, we only consider congestion control based on the blocked-calls-dropped discipline. The resource access priority for handoff calls may be further increased by employing the blocked-calls-queued discipline [12]. The problem discussed in this paper is the following: since the traffic patterns are not uniform, how many channels may be reserved to handoff calls in the different cells?

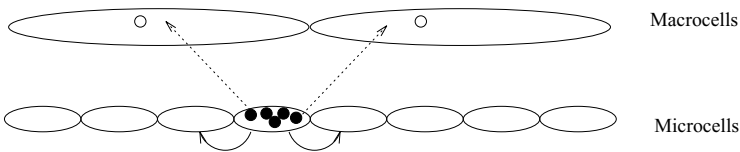
We define  $\Phi(i)$  as the set of the microcell  $i$  neighbours.  $p_{ij}^l$  (resp.  $p_{ij}^h$ ) denotes the probability for a given low speed mobile (resp. high speed mobile) to move from a microcell number  $i$  to a neighbouring microcell number  $j$ . In the different examples, only the case  $p_{ij}^l = p_{ij}^h$  has been considered. In fact, those probabilities might be estimated as a function of the speed of the user. The reservation policies may also depend on those parameters. It should lead to more complex policies ; the optimization of the different thresholds would be much more



complicated. Furthermore, in this paper, only micro/microcell transition probabilities are considered; other reservation schemes may have been designed. Our purpose was mainly to show the influence of reservation schemes based on traffic parameters on the performance of hierarchical cellular networks.

## 2.2 The HPRP Algorithm

In a microcell  $i$ , calls may be connected to the microcell Base Station ( $\mu\_BS(i)$ ) or the corresponding macrocell Base Station ( $M\_BS(i)$ ).  $C_i^\mu$  and  $C_i^M$  are the number of channels in  $\mu\_BS(i)$  and in  $M\_BS(i)$ . Let  $N_i^\mu(t)$  and  $N_i^M(t)$  denote the number of users in microcell  $i$  connected at time  $t$  respectively to  $\mu\_BS(i)$  and to  $M\_BS(i)$ . When the number of occupied channels  $N_i(t) = N_i^\mu(t) + N_i^M(t)$  in a given microcell reaches a threshold  $k$  or a multiple of  $k$  (see Fig. 1), resources are reserved in the neighbours for which the probability of transition is high. We suppose that the number of mobiles in a given micro-cell is managed by a controller of a cluster of cells.  $k$  is a HPRP parameter, it has to be optimized in order to improve the performance of the system. In fact, reservations are made in the corresponding macrocell. No reservation scheme has been considered at the microcell level. This is due to the fact that in the different schemes, the users that may not be accepted at the microcell level may be connected to the corresponding macrocell and that microcell channels are mostly occupied by pedestrian users for which handoff rate is quite low. If  $M\_BS(i)$  has free channels, the reservation takes place immediately. Otherwise, the HPRP algorithm waits for a free channel. A reserved channel corresponds to the potential arrival of a communication. Two intervals are considered: if the transition probability is lower than a given value  $p$ , no reservation are made, else the reservation threshold is  $k$ . The blocked channels can only be used for handoffs. In case of overload (or a state close to the overload), the system does not accept incoming calls but only processes calls in progress.



**Fig. 1.** Principle of HPRP reservation

Ex:  $k = 10$  and  $p = 0$

$N_i(t) = 10 \rightarrow 1^{st}$  reservation (or request for a reservation) in the neighbours of  $i$ .

$N_i(t) = 20 \rightarrow 2^{nd}$  reservation (or request for a reservation) in the neighbours of  $i$ .

Reservations are cancelled when the number of occupied channels becomes lower

than a threshold. When  $N_i(t)$  goes down from 20 to 19 the 2<sup>nd</sup> reservation (or request for a reservation) is cancelled.

### 2.3 Call and Handoff Admission Strategies

Once the frequencies are allocated on micro and macro layers, terminals may be served by the two layers. The question that rises is to know which policy provides the best efficiency. Many strategies were proposed in the literature [13] [10] [6] [2] [7]. Two mobility behaviours are usually considered : pedestrians are quasi-fixed while cars move quickly. In order to maximize the system efficiency, low-speed users are mostly connected to the lower layer while upper layer is used by high-speed users and act as overflow for low layer. In non-reversible systems, a call is never taken back by the lower layer. In reversible systems, the call may be taken back by the microcell layer when a resource is available. In the present paper, three solutions were studied.

**First Scheme (P1).** In the first scheme, when a mobile gets into a new microcell areas, the system tries to connect it to the microcell level. When there is no resource left in the microcell base station, it can be temporarily connected to the macrocell level if there is a free channel (cf. Figure 2).

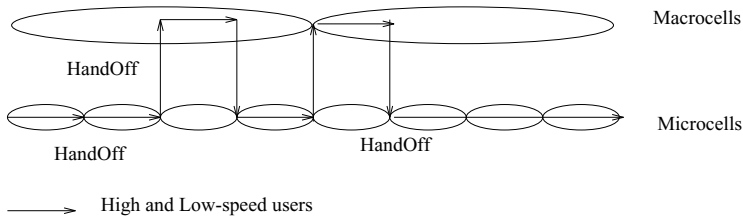
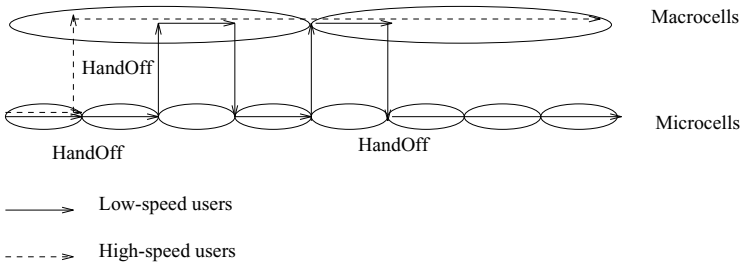


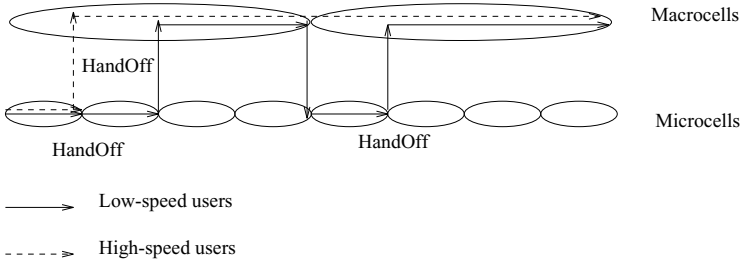
Fig. 2. Admission Policy - First Scheme (P1)

**Second Scheme (P2).** In the second scheme, the previous algorithm is applied to low-speed users. High-speed users try first to access the microcell layer and then overflows to the macrocell if there is no available channel. Once there are connected to the macrocell level they will only experience macro/macro cell handoff and can only be connected to macrocell base stations (see Figure 3.).

**Third Scheme (P3).** The second scheme algorithm is applied to high-speed users. In order to minimize the number of handoffs for low-speed users, when a pedestrian has to be connected to a macrocell base station, it will stay at this level whenever it stays in the same macrocell area. When it enters a new macrocell, it will again access the low-layer level, if there is enough resource (cf. Figure 4.).



**Fig. 3.** Admission Policy - Second Scheme (P2)



**Fig. 4.** Admission Policy - Third Scheme (P3)

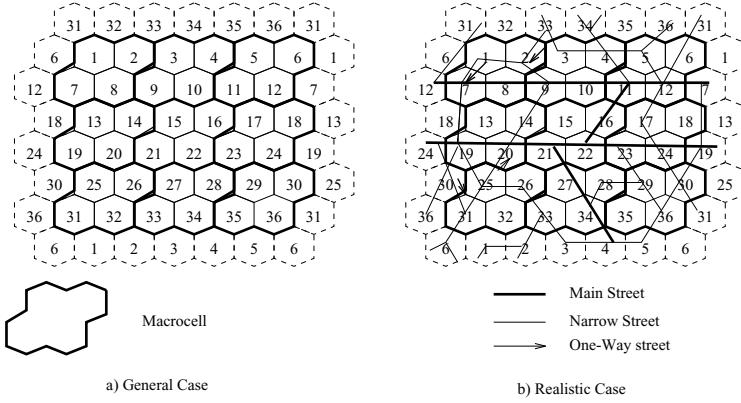
### 3 Simulation model and results

#### 3.1 Simulation Model and Performance Criteria

We simulated the operations of the proposed dynamic guard bandwidth scheme in a mobile wireless system with radio cells having different traffic characteristics. A 36-cell configuration has been chosen. In the first example -General Case- different new call arrival rates and non-uniform transition probabilities are considered. In the second one -Realistic Case- roads, cross-roads, dead-ends, ... are simulated. The routing probabilities are chosen as a function of the topology (cf. Figure 5). For instance,  $p_{7,8}$  and  $p_{7,12}$  are high (main street),  $p_{7,1}$  is nearly zero (one-way street), and new call arrival rates are particularly high in a residential area (cell 13).

Discrete event simulations were run in order to estimate the performance criteria (new call and handoff blocking probabilities and the Grade of Service).

In this paper, classical assumptions are made. We assume that for each time period the traffic is characterized by the arrival of new calls in each microcell and by the transition probabilities of micro/micro handoff calls. The arrival process of new calls is assumed to be Poisson for both low speed (index  $l$ ) and high speed users (index  $h$ ) with respective rates  $\lambda_i^l$  and  $\lambda_i^h$  which depend on the microcell  $i$ . The call duration and the sojourn time of a mobile within a given cell are supposed to have a negative exponential distribution with the parameters  $\mu_c^h$  and



**Fig. 5.** Description of the simulation model

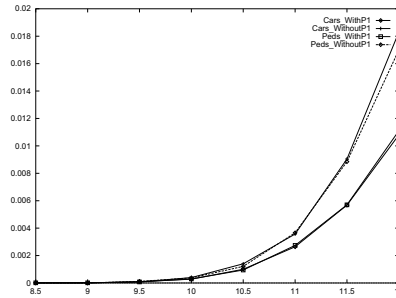
$\mu_s^h$  for the high speed users and  $\mu_c^l$  and  $\mu_s^l$  for the low speed users. In this paper, we shall assume that  $\mu_c^l = \mu_c^h$ .

A useful performance criteria is the Grade of Service (GoS) which can be defined as follows for class  $k$  traffic :

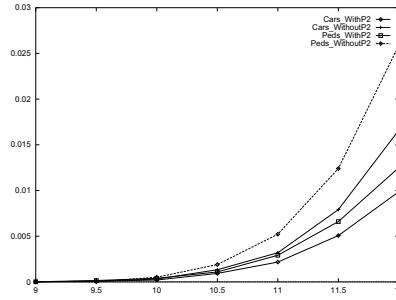
$$GoS^k = P_{nc}^k + \alpha P_{ho}^k. \quad (1)$$

where  $P_{nc}^k$  and  $P_{ho}^k$  denote respectively the new call and the handoff blocking probabilities for class  $k$  users. In order to comply with the QoS constraints of the cellular networks,  $\alpha$  is set to 10 (e.g.  $P_{nc}^k$  should be lower than  $10^{-2}$  and  $P_{ho}^k$  lower than  $10^{-3}$ ).

### 3.2 Results

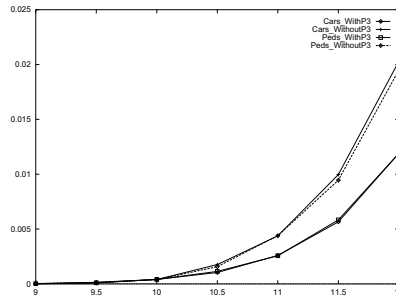


**Fig. 6.** Grade of Service as a function of the traffic load (expressed in Erlang) - First Scheme - General Case



**Fig. 7.** Grade of Service as a function of the traffic load (expressed in Erlang) - Second Scheme - General Case

**Results - General Case.** Let us first present the simulation results in the General Case. In Figures 6, 7 and 8, we represented the GoS as a function of the traffic load (expressed in Erlang).

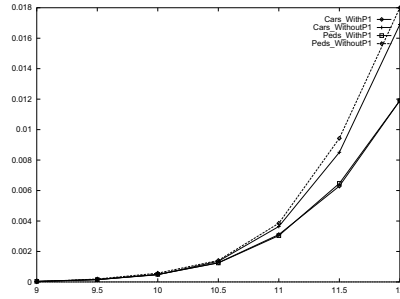


**Fig. 8.** Grade of Service as a function of the traffic load (expressed in Erlang) - Third Scheme - General Case

The number of channels in the microcell level is  $C_i^\mu = 4$  and the number of channels in the macrocell level is  $C_i^M = 80$ . The best performance results were obtained with  $k = 12$  and  $p = 0.35$ .

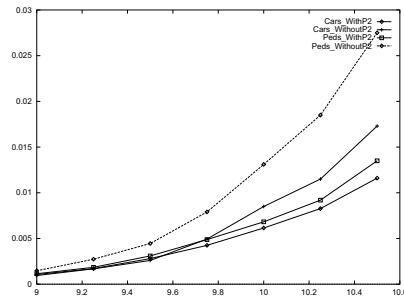
In this general scheme, we wanted to show the influence of the reservation mechanism. In fact, since the number of channels in the lowest level is equal to 4, the different assignment algorithms nearly lead to the same performance results (same order of magnitude). In the three schemes, HPRP really improves the performance of the system. The improvement increases with the traffic rate; for a load of 11.5 Erlangs, it is between 20 % and 40 %. Consequently, it is shown that the reservation principles described below are efficient and may be applied to a more realistic traffic case [1].

**Results - Realistic Case.** In Figures 9, 10 and 11, we represented the GoS as a function of the traffic load (expressed in Erlang). In order to compare the different assignment schemes, the number of channels in the microcell level is  $C_i^{\mu} = 8$  and the number of channels in the macrocell level is  $C_i^M = 64$ .



**Fig. 9.** Grade of Service as a function of the traffic load (expressed in Erlang) - First Scheme - Realistic Case

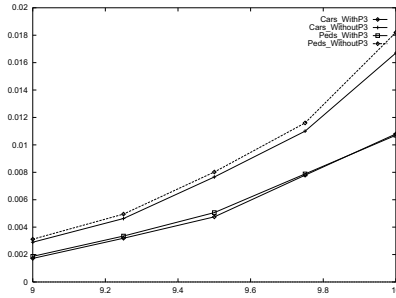
In the first scheme, since the assignment algorithms are the same for both pedestrians and cars, performance results are nearly the same.



**Fig. 10.** Grade of Service as a function of the traffic load (expressed in Erlang) - Second Scheme - Realistic Case

In the second scheme, the number of handoffs of the cars decreases. But, since cars are mostly connected to the macrocell level, they may be blocked even if there is free channels in the microcell level. Consequently, this scheme leads to an underutilization of the bandwidth. In this scheme, pedestrians try to stay at the lowest level as long as possible. So, they may free a macrocell level channel when they get into a new microcell and then they may be blocked during the next handoff even in the same macrocell. It finally leads to a great difference

between the GoS for the two kinds of traffic.



**Fig. 11.** Grade of Service as a function of the traffic load (expressed in Erlang) - Third Scheme - Realistic Case

In the last case, the bandwidth utilization is worse than in the previous case because pedestrians stay longer at the highest level. The GoS is nearly the same for both low-speed and high-speed users.

The first scheme leads to the best performance results but introduces many handoffs, especially for high-speed users. The last two policies reduce the number of handoffs but the channels of the microcell layer are underloaded and consequently lead to bad performance results. The last policy is easy to implement in a realistic system but the number of microcell channels has to be low. The best performance results of HPRP are obtained with the same threshold values. In this case, HPRP strongly improves the GoS in the different assignment algorithms. For a GoS near 0.01, the improvement is about 30% in the three cases. It is shown that the relative improvement depends on the admission policy and on the local topology. But, in all the different cases, the use of HPRP reveals itself efficient.

## 4 Conclusion

In this paper, a new guard channel algorithm has been proposed: HPRP. The required guard bandwidth can be calculated off line and loaded into lookup tables to facilitate the dynamic allocation. This mechanism has been implemented in a hierarchical architecture of micro/macrocels. Three call and handoff strategies were proposed for both low-speed and high-speed users. Two configurations were simulated : in the first one -general case- non-uniform traffic conditions were considered, in the second one -realistic case- a given topology of roads, dead-ends ... has been chosen. In all the considered examples, HPRP leads to an improvement of nearly 30% of the grade of service.

It is shown that if some cell to cell handoff probabilities are known (even approximately), it is possible to use them in order to manage the bandwidth efficiently.

Adaptive reservation schemes prove to be efficient, especially when the traffic is unbalanced, which seems to be a realistic assumption.

## References

1. A.-L. Beylot, S. Boumerdassi and G. Pujolle, A New Prioritized handoff Strategy Using Channel Reservation in Wireless PCN, IEEE GLOBECOM'98, Sydney, Australia, November 1998.
2. G.L. Choudury, S.S. Rappaport, Cellular Communication Schemes Using Generalized Fixed Channel Assignment and Collision Type Request Channels, IEEE Transactions on Vehicular Technology, Vol VT-31, pp. 53-65, May 1982.
3. J. Daigle and N. Jain, A queuing system with two arrival streams and reserved servers with application to cellular telephone, IEEE INFOCOM'92, April 1992, pp. 2161-2167.
4. R. Guerin, Queueing-blocking system with two arrival streams and guard channels, IEEE Transactions on Communications, Vol COM-36, pp. 153-163, February 1988.
5. D. Hong and S.S. Rappaport, Traffic model and performance analysis for cellular mobile radio telephone systems with prioritized and non-prioritized handoff procedures, IEEE Transactions on Vehicular Technology, Vol VT-35, pp. 77-92, August 1986.
6. L.-R. Hu and S.S. Rappaport, Personal Communication Systems Using Multiple Hierarchical Cellular Overlays, IEEE Journal on Selected Areas in Communications, Vol SAC-13, n2, pp. 406-415, February 1995.
7. X. Lagrange, Multitier Cell Design, IEEE Communications Magazine, pp. 60-64, August 1997.
8. L. Ortigoza-Guerrero and A.H. Aghvami, On the Optimum Spectrum Partitioning in a Microcell/Macrocell Cellular Layout with Overflow, IEEE GLOBECOM'97, Phoenix, USA, 1997.
9. D. Raychoudhuri and A.D. Wilson, ATM-Based Transport Architecture for Multiservice wireless personal Communication Networks, IEEE Journal on Selected Areas in Communications, Vol SAC-12, n8, October 1992.
10. M. Naghshineh and M. Schwartz, Distributed call admission control in mobile/wireless networks, IEEE Journal on Selected Areas in Communications, Vol SAC-14, pp. 711-717, May 1996.
11. L.-C. Wang, G.L. Stuber and C.-T. Lea, Architecture Design, Frequency Planning and Performance Analysis for a Microcell/Macrocell Overlaying System, IEEE Transactions on Vehicular Technology, Vol VT-46, n4, pp. 836-848, November 1997.
12. C.H. Yoon and C.K. Un, Performance of personal portable radio telephone systems with and without guard channels, IEEE Journal on Selected Areas in Communications, Vol SAC-11, pp. 911-917, August 1993.
13. K.L. Yeung and S. Nanda, Channel Management in Microcell/Macrocell Cellular Radio Systems, IEEE Transactions on Vehicular Technology, Vol VT-45, n4, pp. 601-612, November 1996.
14. O.T.W. Yu and V.C.M. Leung, Adaptive Resource Allocation for Prioritized Call Admission over an ATM-Based Wireless PCN, IEEE Journal on Selected Areas in Communications, Vol SAC-15, n7, pp. 1208-1225, September 1997.



# Dynamic IEEE 802.11: Design, Modeling and Performance Evaluation

Federico Cali, Marco Conti, and Enrico Gregori

CNUCE, Institute of National Research Council

Via Alfieri 1 - 56127 Pisa - Italy

{Marco.Conti, Enrico.Gregori}@cnuce.cnr.it

**Abstract.** In Wireless LANs (*WLANs*) the medium access control (*MAC*) protocol is the main element that determines the efficiency of sharing the limited communication bandwidth of the wireless channel. Previous papers have shown that an appropriate tuning of the backoff algorithm can drive the IEEE 802.11 protocol close to its theoretical limits. In this work we analytically study the performance of the IEEE 802.11 protocol with a dynamically tuned backoff. Specifically, we investigate the sensitiveness of the dynamically tuned backoff algorithm to some network configuration and traffic parameters. Our results indicate that, under stationary traffic conditions, the capacity of the enhanced protocol approaches the theoretical upper bound value in all the configurations analyzed. Furthermore, we also show that the algorithm quickly re-tunes the backoff window size when the network traffic conditions change thus guaranteeing, even with non-stationary traffic conditions, a capacity that is very close to the optimal value. Robustness of the protocol to error conditions is also evaluated. In the paper it is shown the protocol correctly reacts to error conditions.

## 1 Introduction

For decades Ethernet has been the predominant network technology for supporting distributed computing. In recent years the proliferation of portable and laptop computers has led to LAN technology being required to support wireless connectivity ([6], [9]). In addition to providing for computer mobility, Wireless LANs (*WLANs*) are easier to install and save the cost of cabling. The success of WLANs is connected to the development of networking products that can provide wireless network access at a competitive price. A major factor in achieving this goal is the availability of appropriate networking standards. In this paper we focus on the IEEE 802.11 standard for WLANs [8].

The design of wireless LANs has to concentrate more on bandwidth consumption than wired networks. This is because wireless networks deliver much lower

---

Work carried out under the financial support of CNR, in the framework of the project “Sistemi radiomobili multimediali nell’evoluzione verso UMTS”.

bandwidths than wired networks, e.g. 1-2 Mbps vs. 10-150 Mbps [11]. Since a WLAN relies on a common transmission medium, the transmissions of the network stations must be coordinated by the Medium Access Control (MAC) protocol. The fraction of channel bandwidth used by successfully transmitted messages gives a good indication of the overheads required by the MAC protocol to perform its coordination task among stations. This fraction is known as the utilization of the channel, and the maximum value it can attain is known as the *capacity* of the MAC protocol ([10], [5]). Previous works have shown that an appropriate tuning of the IEEE 802.11 backoff algorithm can significantly increase the protocol capacity ([1], [2], [1314]). In [1] the authors propose to tune the backoff window size on the number of active stations, this number being estimated by observing the channel status. Weinmiller et al. [13] outlined a way to modify the backoff distribution to uniformly spread the channel access in a contention window and thus decrease the collision probability. Both studies use simulative analyses to show that significant improvements in protocol capacity can be achieved by modifying the backoff algorithm. In [2] the theoretical upper limit for the IEEE 802.11 protocol capacity was derived. Furthermore, it was shown that with an appropriate tuning of the IEEE 802.11 backoff window size, the protocol capacity can approach the theoretical capacity bound. The results indicate that the optimal backoff window size very much depends on the traffic conditions. Hence the optimal protocol capacity can only be achieved if the backoff window is dynamically tuned at run-time following the evolution of the network configuration (i.e. number of active stations) and load traffic conditions (i.e. average length of transmitted messages). In a previous work we have proposed and analyzed a distributed algorithm to dynamically tune the backoff window-size of the IEEE 802.11 MAC protocol to take into consideration the (dynamically changing) load traffic conditions [Cal 99]. As far as the network configuration is concerned, a fixed network configuration (number  $M$  of active stations) is assumed. A performance analysis indicated that, when the number of active stations is not very far from the assumed  $M$  value (i.e. in the range between half and the double of the assumed  $M$  value) the efficiency of the protocol remains very close to its theoretical bounds. However, by further increasing the distance between  $M$  and the real number of active stations the efficiency of the dynamic IEEE 802.11 protocol further degrades thus making it necessary to introduce mechanisms for a dynamic estimation of the  $M$  value (see for example [1], [2]). In this work, by exploiting some ideas presented in [2] we extend (and evaluate) the distributed algorithm to dynamically tune the backoff window-size of the IEEE 802.11 MAC protocol to cope also with a dynamically changing number of active stations. Specifically, the proposed algorithm is executed independently by each station that by observing the channel status derives an estimate of both the network and load configurations. Then, by exploiting this information, a station computes the optimal backoff window size for the current configuration. In the following we named *Dynamic IEEE 802.11* the IEEE 802.11 MAC protocol extended with such an estimation-based backoff algorithm. By developing a Markovian model of the Dynamic IEEE 802.11 protocol we extensively analyze the properties of the enhanced protocol. Specifically, we study the algorithm's behavior both under steady-state and transient conditions. Furthermore, we analyze the robustness of the

protocol, that is the protocol ability to cope with errors in the estimations of the network and load configurations. Errors can be induced by the unreliable wireless medium.

The paper is organized as follows. Section 2 presents the IEEE 802.11 MAC protocol and the dynamically tuned backoff algorithm. In Sections 3 and 4 we study the protocol behavior in steady-state and in transient conditions, respectively. Finally in Section 5 we study the protocol robustness.

## 2 IEEE 802.11 MAC Protocol

The IEEE 802.11 MAC layer protocol provides asynchronous, time-bounded, and contention free access control on a variety of physical layers. The basic access method in the IEEE 802.11 MAC protocol is the *Distributed Coordination Function* (DCF) which is a *Carrier Sense Multiple Access with Collision Avoidance* (CSMA/CA) MAC protocol [8].

The model used in this paper to evaluate the protocol performance figures does not depend on the technology adopted at the physical layer (e.g. infrared and spread spectrum). However, the physical layer technology determines some network parameter values, e.g. SIFS, DIFS, and backoff slot time. Whenever necessary, we choose the values of these technology-dependent parameters by referring to the frequency-hopping-spread-spectrum technology at a 2 Mbps transmission rate.

### 2.1 Theoretical Capacity Limits of the IEEE 802.11 Protocol

In [2] the efficiency of the IEEE 802.11 standard for wireless LANs was investigated in depth. Specifically, by deriving an analytical formula for the protocol capacity: *i*) the theoretical upper bound of the IEEE 802.11 protocol capacity was identified, and *ii*) it was shown that, depending on the network configuration, the standard may operate very far from the theoretical limits. A summary of these results is reported in Figure 1 that compares, for several network configurations, the IEEE 802.11 capacity with the analytical bounds. These results have been obtained with the configuration parameter values reported in Table 1.

**Table 1.** WLAN configuration

Parameter	value
SIFS	28 $\mu$ sec
DIFS	128 $\mu$ sec
backoff slot time	50 $\mu$ sec
bit rate	2 Mbps
propagation delay	1 $\mu$ sec
stations	10, 50, 100
CWmin	8
CWmax	256

The results show that for almost all configurations the IEEE 802.11 capacity can be improved significantly. As highlighted by the figure, the distance between the IEEE 802.11 and the analytical bound increases with the number,  $M$ , of active stations. The figure also shows an additional curve tagged as optimal window size. This curve represents the capacity of an IEEE 802.11 protocol but with a single contention window whose size is equal to the optimal value (for that network and traffic configuration) that is computed from the formulas developed for the theoretical upper bound of the capacity [2].

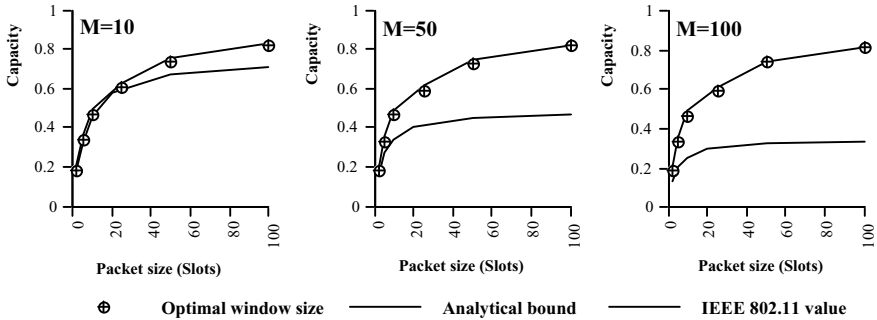


Fig. 1. Analytical bounds vs. IEEE 802.11 capacity

## 2.2 Dynamic IEEE 802.11

The results presented in the previous section indicate that the IEEE 802.11 protocol often operates very far from its theoretical limits and that the theoretical capacity limits can be achieved with an adequate tuning of the backoff window size. Based on these observations, in this paper we investigate the Dynamic IEEE 802.11 protocol, i.e. the IEEE 802.11 protocol with an algorithm for the dynamic tuning of the backoff parameters.

The results presented in [2] indicate that, given a network and traffic configuration, there exists an optimal value for the average backoff interval  $E[B]$ . This optimal value can be easily mapped on the value of the parameter  $p$  of a geometric distribution with an average  $E[B]$ . Hereafter  $p_{\min}$  will indicate the  $p$  value corresponding to the optimal backoff interval.

The advantage of using a geometrically distributed backoff with respect to the uniformly distributed backoff of the IEEE 802.11 is that by simply changing the  $p$  parameter value we can immediately adapt the backoff window size to a change in the network or traffic conditions.

The Dynamic IEEE 802.11 protocol is similar to a  $p$ -persistent protocol [11]: at the beginning of an empty slot a station transmits (in that slot) with a probability  $p$ . The differs With a probability  $1-p$  the transmission is deferred. This procedure is repeated whenever an empty slot is detected on the channel. The main differences between and a classical  $p$ -persistent protocol are:

- in a classical  $p$ -persistent protocol the value of the  $p$ -parameter is constant while in the Dynamic IEEE 802.11 protocol the  $p$  value changes depending on the network configuration and load conditions;
- in a classical  $p$ -persistent protocol the length of the backoff interval is independent of the status of the channel during the backoff itself, while in the Dynamic IEEE 802.11 protocol, as in the standard IEEE 802.11 protocol, the backoff decreases only when the channel is idle.

The main element of the Dynamic IEEE 802.11 protocol, with respect to the standard one is the algorithm that is in charge to dynamically adjust the  $p$ -value to the network and load conditions. In [3] it was proposed and evaluated an algorithm that maximizes the protocol capacity by dynamically adapting the  $p$ -value to the load configuration. However, the  $p$ -estimation algorithm proposed in [3] assumed that each station knew the number  $M$  of the active stations in the network a priori. This is a strong assumption as, in the real network, the number of active stations varies considerably.

Here, we propose and analyze a dynamic backoff algorithm that does not require any a priori knowledge on the network and load conditions. Specifically the algorithm, by observing the channel status, estimates at run-time both the number of active stations in the network and the load configuration. By exploiting this information the algorithm updates the  $p$ -value. The updates of the  $p$ -value occur at the end of very *virtual transmission time*, where a virtual transmission time is the time interval between two consecutive transmission attempts.

Before presenting the procedure used by the Dynamic IEEE 802.11 protocol to estimate the load and network configurations, it is useful to better identify the information required by the protocol to compute the optimal  $p$ -value.

**Optimal  $p$ -value.** In this section we define a criteria to compute the  $p$ -value to guarantee that the Dynamic IEEE 802.11 protocol achieves the theoretical protocol capacity, i.e. the maximum throughput in the current load and network configuration.

In [2], by exploiting the geometric backoff assumption, it has been shown that the protocol capacity can be expressed as:

$$\rho_{\max} = \frac{\bar{m}}{E[t_{\text{success}}]}$$

where  $E[t_{\text{success}}]$  is the average temporal distance between two consecutive successful transmissions, and  $\bar{m}$  is the average message length.

Furthermore, for a given network and load configuration,  $E[t_{\text{success}}]$  is a function of the  $p$ -value only. Hence, the value of  $p$  that minimizes  $E[t_{\text{success}}]$ , say  $p_{\min}$ , guarantees the maximum protocol capacity. Since the exact  $p_{\min}$  derivation is expensive from a

computational standpoint, in [2], it was proposed to approximate  $p_{\min}$  with the  $p$  value that satisfies the following relationship:<sup>1</sup>

$$E[Coll]_{|Collision} \cdot E[N_c] = (E[N_c] + 1) \cdot E[Idle\_p] \cdot t_{slot} \quad (1)$$

where  $E[Coll]_{|Collision}$  is the average collision length,  $E[N_c]$  is the average number of collisions between two consecutive successful transmissions, and  $E[Idle\_p]$  is the average number of consecutive idle slots.

By noting that  $E[N_c] / (E[N_c] + 1)$  is the probability that a collision occurs given a transmission attempt, Equation (1) can be written as:

$$E[Coll]_{|collision} = \frac{E[Idle\_p] \cdot t_{slot}}{p_{collision}} \Rightarrow E[Coll] = E[Idle\_p] \cdot t_{slot} \quad (2)$$

where  $p_{collision} = E[N_c] / (E[N_c] + 1)$ , and  $Coll$  is the time the channel is busy due to a collision given that a transmission attempt occurs, also referred to as *collision cost*. Obviously,  $Coll$  is equal to zero if the transmission attempt is successful, otherwise it is equal to the collision length.

Analytical formulas for the quantities of Equation (2) are defined in LEMMA 1 whose proof can be found in [2].

LEMMA 1. Assuming that for each station *i*) the backoff interval is sampled from a geometric distribution with parameter  $p$ , and ii) packet lengths are i.i.d. sampled from a geometric distribution with parameter  $q$ :

$$\begin{aligned} E[N_c] &= \frac{1 - (1-p)^M}{Mp(1-p)^{M-1}} - 1 \\ E[Coll]_{|collision} &= \frac{t_{slot}}{1 - \left[ (1-p)^M + Mp(1-p)^{M-1} \right]} \cdot \left[ \sum_{h=1}^{\infty} \left\{ \left[ (1-pq^h)^M - (1-pq^{h-1})^M \right] \right\} \cdot \frac{Mp(1-p)^{M-1}}{1-q} \right] \\ E[Idle\_p] &= \frac{(1-p)^M}{1 - (1-p)^M} \end{aligned}$$

As it clearly appears, from LEMMA 1, the quantities of Equation (2) can be expressed as a function of  $p$ . Hence, Equation (2) provides the criteria to identify the  $p$ -value that maximizes the protocol capacity. Specifically, by observing the channel status, a station after each transmission attempts updates its estimate of the average collision cost,  $E[Coll]$ . Hence, if the station has an estimate of the number of active stations,  $M$ , by exploiting the formula

$$E[Idle\_p] = \frac{(1-p)^M}{1 - (1-p)^M} \quad ,$$

<sup>1</sup> A similar approximation of the optimal point was proposed in [7] for an Aloha CSMA protocol.

it can compute the value of  $p$  that satisfies the optimal criteria defined by Equation (2).

To summarize, a station to implement the dynamic backoff algorithm needs the knowledge of  $M$  or at least of an estimate of it, say  $Me$ ,  $E[Coll]$  and  $E[Idle\_p]$ . In the next section we show how a station can obtain these information by manipulating the information related to the channel status.

**Run-time estimates.** As stated in the previous section a station modifies the  $p$  value that it uses for the backoff algorithm at the end of each virtual transmission interval (i.e. after each transmission attempt).

To better clarify the operations performed by a station let us refer to Figure 2. Specifically, the figure represents a station behavior during the  $(n+1)$ th virtual transmission interval by assuming that at the beginning of that interval, i.e. the end of the  $n$ -th virtual transmission interval, it has the following information:

- $p_n$  is the optimal value of  $p$  ;
- $Me_n$  is the estimated number of active stations;
- $E[Idle\_p]_n$  is the average number of consecutive empty slots;
- $E[Coll]_n$  is the average collision cost.

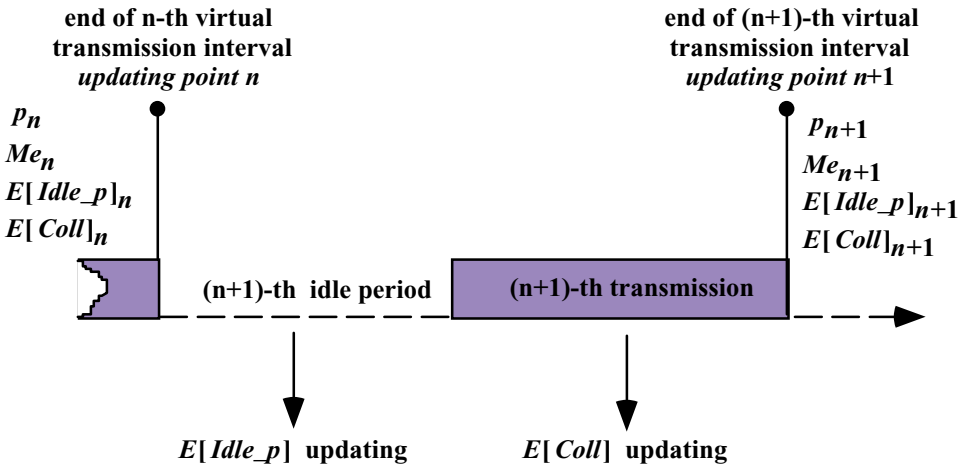


Fig. 2. Estimates updating

Each station by using the carrier sensing mechanism can observe the channel status<sup>2</sup> and measure the length of both the last idle period and the last transmission attempt.

<sup>2</sup> In a CSMA protocol a station observes all the channel busy periods. A busy period is assumed to be a collision if an ACK does not immediately follow.

From this two values, the algorithm, by exploiting a moving averaging window, approximates the average idle period length and the average collision length:

$$\begin{aligned} E[Idle\_p]_{n+1} &= \alpha \cdot E[Idle\_p]_n + (1 - \alpha) \cdot Id_{n+1} \\ E[Coll]_{n+1} &= \begin{cases} \alpha \cdot E[Coll]_n + (1 - \alpha) \cdot Ta_{n+1} & \text{if collision occurs} \\ \alpha \cdot E[Coll]_n & \text{if success occurs} \end{cases} \end{aligned} \quad (3)$$

where  $E[Idle\_p]_{n+1}$  and  $E[Coll]_{n+1}$  are the approximations of  $E[Idle\_p]$  and  $E[Coll]$  respectively at the end of the  $(n+1)$ th transmission attempt,  $Id_{n+1}$  is the length of the  $(n+1)$ th idle period,  $Ta_{n+1}$  is the length of the  $(n+1)$ th transmission attempt and  $\alpha$  is a smoothing factor. The use of a smoothing factor,  $\alpha$ , is widespread in the networking environment to obtain reliable estimates from the network estimates by avoiding harmful fluctuations e.g., RTT estimation in TCP [12].

It is worth noting that  $E[Idle\_p]_{n+1}$  is estimated by observing the channel status, hence its value is function of the  $p$ -value used by the stations and the real number  $M$  of active stations. Hence, from the knowledge of  $E[Idle\_p]_{n+1}$  and the  $p$ -value a station can derive an estimate of the number of active stations,  $Me_{n+1}$ . Specifically, at the end of the  $(n+1)$ th virtual transmission interval each station computes an estimate of  $M$ , say  $M_{comp}$ , by exploiting  $E[Idle\_p]_{n+1}$  and the formula defined in Lemma 1 that expresses  $E[Idle\_p]$  as a function of  $p$  and  $M$ :

$$M_{comp} = \frac{\ln\left(\frac{E[Idle\_p]_{n+1}}{E[Idle\_p]_{n+1} + t_{slot}}\right)}{\ln(1 - p_n)} \quad (4)$$

Finally, from (4), the new estimate of the number of active station  $Me_{n+1}$  is computed by:

$$Me_{n+1} = \alpha \cdot M_{comp} + (1 - \alpha) \cdot Me_n \quad (5)$$

The updated estimate of the number of active stations  $Me_{n+1}$  is then used together with  $E[Coll]_{n+1}$  (see (3)) to compute the value of  $p$  that is optimal (i.e. maximizes the protocol capacity) for the new network and load conditions. Specifically, according to Equation (2), the optimal  $p$ -value should guarantee a balance between the average collision cost and the average idle-period length. Each station, by exploiting Lemma 1, expresses the average idle-period length as a function of  $p$  and of the number of active stations (note that a station does not have the knowledge of this number but has an estimate of it,  $Me_{n+1}$ ). Then, by using its estimate of  $E[Coll]_{n+1}$  computes the new optimal value of  $p$ ,  $p_{n+1}$ , from the following formula:

$$p_{n+1} = 1 - Me_{(n+1)} \sqrt{\frac{E[Coll]_{n+1}}{E[Coll]_{n+1} + t_{slot}}} \quad (6)$$



The feasible range of  $p$  values has 1 as its upper bound (a station can use all the channel bandwidth when it is alone). The lower bound is set to the optimal  $p$  value for the maximum number of station allowed in the network (e.g. 100 or 500).

In [2] it was shown that, in a IEEE 802.11 WLAN, if each station tunes its backoff algorithm to the optimal  $p$ -value for the current network and load configuration, the MAC protocol capacity is very close to its theoretical bound. In this work we have presented an algorithm which first estimates the current network and load configuration and then from this knowledge it tunes its backoff algorithm. Hence, in this case the knowledge a station has about the network and load configuration is not exact but depends on an estimation process. In the next sections, by exploiting the Markovian model developed in [4] we study the behavior of the dynamic backoff-tuning algorithm. Specifically, we investigate the protocol performance both in steady-state and transient conditions. In addition, we investigate the robustness of the protocol to possible errors during the estimation process. These errors can occur as the estimation process is based on the carrier sensing of the channel, and it is well known that wireless channels are unreliable.

### 3 Dynamic IEEE 802.11: Protocol Behavior in Steady State Conditions

Table 2 compares, for two network ( $M=10, 20$ ) and traffic ( $q=0.5, 0.99$ ) configurations, the protocol capacity values obtained with our analytical model with the theoretical upper bounds derived in [2].<sup>3</sup> Furthermore, we analyze the impact of the  $\alpha$  smoothing value ( $\alpha=0.5, 0.90, 0.99$ ) on the steady-state behavior of the Dynamic IEEE 802.11 protocol.

**Table 2.** Dynamic IEEE 802.11 Protocol capacity ( $M=10$ )

	$\bar{m} = 100$ slots ( $q = 0.99$ )			$\bar{m} = 2$ slots ( $q = 0.50$ )		
$\alpha=0.50$	13.09	0.00831	0.8018	13.36	0.0427	0.2004
$\alpha=0.90$	10.33	0.01058	0.8220	9.17	0.0685	0.2009
$\alpha=0.99$	9.78	0.01192	0.8237	10.16	0.0559	0.2081
IEEE			0.7135			0.1818
802.11						
ideal	10	0.01150	0.8257	10	0.0525	0.2088
values						

<sup>3</sup> The analytical results were obtained with  $\alpha=0.9$ , and 20 states in the set  $I$ .

**Table 3.** Dynamic IEEE 802.11 Protocol capacity ( $M=20$ )

	$\bar{m} = 100$ slots ( $q = 0.99$ )			$\bar{m} = 2$ slots ( $q = 0.50$ )		
$\alpha=0.50$	27.14	0.00388	0.7972	27.41	0.0210	0.1990
$\alpha=0.90$	18.57	0.00517	0.8126	17.58	0.0282	0.1985
$\alpha=0.99$	19.00	0.00603	0.8214	19.98	0.0279	0.2057
IEEE 802.11 ideal values			0.6113			0.1789
	20	0.00572	0.8223	20	0.0279	0.2060

The results show that the dynamic tuning algorithm is very effective for the network and traffic configuration analyzed. As shown by the analytical results presented in Tables 2 and 3, the capacity of a WLAN implementing the dynamic-backoff tuning algorithm is always very close to the theoretical capacity upper bound (see the ideal-value line in the two tables). Furthermore, the tables show, as expected, the impact of the  $\alpha$  smoothing value. As we are investigating the protocol behavior in stationary conditions, with the increase of the  $\alpha$  values the statistical fluctuations of the quantities estimated by observing the status of the channel becomes less relevant and thus the idle-period and collision-cost estimates are always very close to their steady-state average values. Our results indicate that  $\alpha=0.50$  is not appropriate and  $\alpha=0.99$  is the best choice for a system operating in stationary conditions. However,  $\alpha=0.90$  provides statistics that are quite close to the ideal values as well, and we can expect that  $\alpha=0.90$  is more appropriate when the load and/or network conditions changes because it potentially reduces the length of transient phases.

#### 4 Dynamic IEEE 802.11: Protocol Behavior in Transient Conditions

In this section we analyze the protocol promptness to re-tuning the backoff parameters when the network state sharply changes. Results presented in Table 4 are obtained as follows: the network is assumed to be in steady state with 2 active nodes (20 active nodes). This means that at time 0 we assume that the estimated  $M$  is 2 (20) and the  $p$  value used for the backoff algorithm is the theoretically-optimal  $p$  value for  $M=2$  ( $M=20$ ). At time  $0^+$  the number of active nodes becomes 10. Exploiting our analytical model we evaluate the average first passage time to the new steady state, i.e., the time to update the  $M$  estimate to 10. This is done through the standard procedure for the computation of the first-passage-time distribution in a Markov chain. At the step 0 of the procedure, the system is with probability one in the optimal state for  $M=2$ .

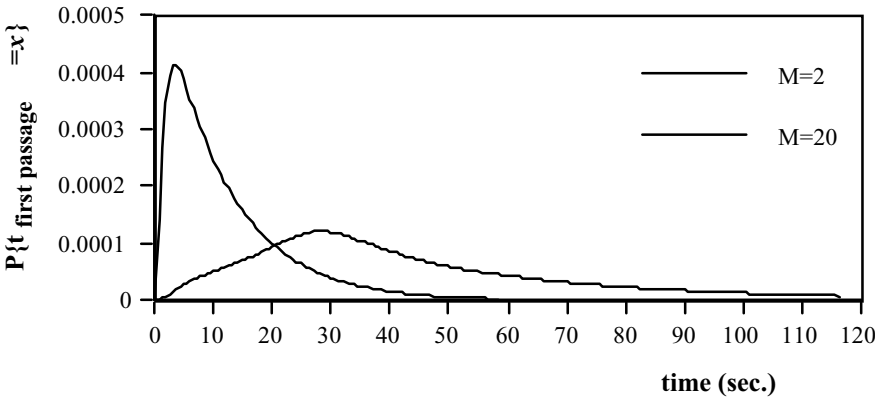
**Table 4.** Average first-passage time to the new steady state

starting state	M=2, $p_{opt}(2)$	M=20, $p_{opt}(20)$
$\alpha=0.5$	9.2 sec.	6.8 sec.
$\alpha=0.9$	40 sec.	12 sec.
$\alpha=0.95$	231 sec.	84 sec.
$\alpha=0.99$	4000 sec.	2090 sec.

Table 4 presents the average first passage time for various  $\alpha$  values. These first passage times remain quite short for smoothing factor  $\alpha$  up to 0.9. Increasing further the  $\alpha$  value makes the transient phase significantly longer. The minimum transient is obviously obtained with  $\alpha=.5$ , but as shown in the steady-state analysis (see Section 4),  $\alpha=.5$  makes our dynamic algorithm too tied to the fluctuations of the network estimates, thus reducing the protocol capacity.

Results presented in Table 4 and in Section 4 indicate that  $\alpha=.9$  is a good compromise between precision and promptness.

The difference between the average first-passage time from  $M=2$  to  $M=10$ , and from  $M=20$  to  $M=10$  can be explained by noting that in the former case the estimated  $M$  has to increase five times, while in the latter case it is reduced to its half.



**Fig. 3.** First passage time distribution

To better analyze the first-passage-time statistics, in Figure 3 we plot the steady-state distribution of the first passage time for  $\alpha=.9$ . The figure indicates that transient intervals are, in the worst case, about one or two minutes. These values are not critical also because it is very unlikely that such a sharp change in the traffic profile occurs in a realistic scenario.

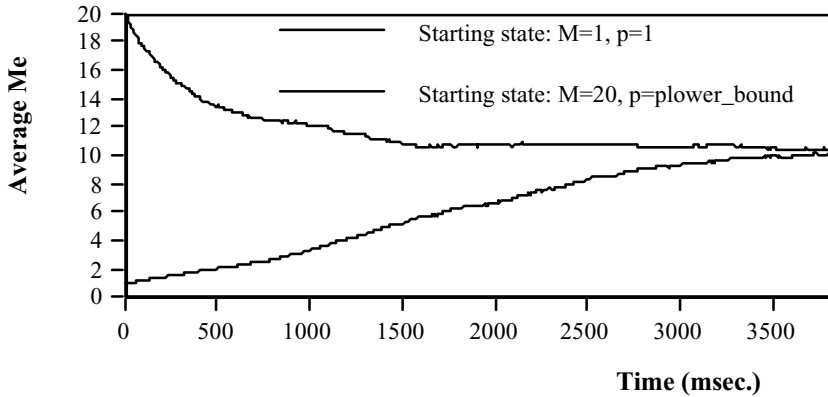


Fig. 4. System behavior when the starting state is wrong

## 5 Dynamic IEEE 802.11: Protocol Robustness to Estimation Errors

In this subsection we discuss the robustness of the protocol. We assume that due to some errors in the estimation phase or biasing induced by the hidden-station phenomenon all the  $M$  active stations assume  $Me=1$  and thus  $p=1$ . Thus a collision will immediately occur and, as a result of the collision, the  $p$  parameter value used in the next backoff is much smaller (0.0853). From this time onward the evolution is probabilistic. Figure 4 plots the average  $Me$  estimate. This estimate is computed using standard transient-analysis method for markovian systems. The Markov chain used for this analysis corresponds to a 10 active stations scenario, i.e. ( $M=10$ ). The figure indicates that the system correctly reacts to the wrong estimate and, in few seconds, the estimates for  $M$  and then for  $p$  become the optimal ones.

On the other hand, wrongly assuming a highly-congested network ( $M=20$  and  $p=\text{minimum value}$ ) when  $M=10$  is less critical. The evolution of the  $Me$  is probabilistic and thus its study requires a transient Markov chain analysis. The average  $Me$  plotted in Figure 4 indicates that the system correctly reacts to the wrong estimate in this case too and after a few seconds the correct  $Me$  value is reached.

## References

- [1] G.Bianchi, L.Fratta, M. Oliveri, "Performance Evaluation and Enhancement of the CSMA/CA MAC Protocol for 802.11 Wireless LANs", *Proceedings of PIMRC 1996*, October 1996, Taipei, Taiwan, pp. 392-396.
- [2] F.Cali, M. Conti, E. Gregori, "IEEE 802.11 Wireless LAN: Capacity Analysis and Protocol Enhancement", *Proc. INFOCOM 98*, San Francisco, March 1998.
- [3] F.Cali, M. Conti, E. Gregori, "Performance Modeling of an Enhanced IEEE 802.11 protocol", *Proc. IFIP ATM 99*, June 1999, Antwerp, Belgium.

- [4] F.Cali, M. Conti, E. Gregori, "IEEE 802.11 protocol: design and performance evaluation of an adaptive backoff mechanism" CNUCE report, September 1999.
- [5] M. Conti, E. Gregori, L. Lenzini, "Metropolitan Area Networks" *Springer Verlag*, November 1997.
- [6] G.H. Forman, J. Zahorjan, "The challenges of mobile computing", *IEEE Computer*, April 94, pp.38-47.
- [7] R.G. Gallager, "A perspective on multiaccess channels", *IEEE Trans on Information Theory*, Vol.31, 1985, pp. 124-142.
- [8] IEEE standard for Wireless LAN- Medium Access Control and Physical Layer Specification, P802.11, November 1997.
- [9] T. Imielinsky, B.R. Badrinath, "Mobile Computing: Solutions and Challenges in Data Management", *Communications of ACM*, Oct. 94.
- [10] J.F. Kurose, M. Schwartz, Y. Yemini, "Multiple access protocols and time constraint communications", *ACM computing Surveys*, Vol. 16, pp.43-70.
- [11] W. Stallings, "Local & Metropolitan Area Networks", *Prentice Hall*, 1996.
- [12] W.R.Stevens, "TCP/IP Illustrated, Volume 1: The Protocols", *Addison-Wesley*, Reading, MA, 1994
- [13] J. Weinmiller, H. Woesner, JP Ebert, A. Wolisz, "Analyzing and Tuning the Distributed Coordination Function in the IEEE 802.11 DFWMAC Draft Standard", *Proceedings of MASCOT '96*, February 1996, San Jose, California.
- [14] J. Weinmiller, M. Schläger, A. Festag, A. Wolisz, "Performance Study of Access control in Wireless LANs-IEEE 802.11 DFWMAC and ETSI RES 10 HIPERLAN", *Mobile Networks and Applications*, Vol. 2, 1997, pp.55-67.

# TCP/IP over the Bluetooth Wireless Ad-hoc Network

Niklas Johansson, Maria Kihl and Ulf Körner

Department of Communication Systems, Lund University, Sweden  
(niklasj, maria, ulfk)@telecom.lth.se

**Abstract.** Bluetooth is a wireless ad-hoc network concept that was presented in February 1998 by its five original promoters Ericsson, Nokia, IBM, Toshiba and Intel. With Bluetooth, mobile terminals within range of each other can set up ad-hoc connections for both synchronous traffic, e.g. voice, and asynchronous traffic, e.g. IP-based data traffic. In this paper we analyse how well Bluetooth can carry TCP/IP traffic and in particular we show that though the radio channel is very disturbed the TCP Vegas protocol with its flow control mechanism can be carried very well. With ARQ handled at the Bluetooth level, retransmissions are made immediately after a packet error and thus the delays, normally introduced are kept acceptably short. In our model important mechanisms in TCP Vegas as well as Bluetooth mechanisms are modelled in detail and we show that TCP throughput is quite high and delays are kept short for packet error probabilities up to 50% and moderate loads.

## 1. Introduction

Bluetooth is a wireless ad-hoc network concept that was presented in February 1998. With Bluetooth, mobile terminals within range of each other can set up ad-hoc connections. It is designed for both synchronous traffic, e.g. voice, and asynchronous traffic, e.g. IP-based data traffic. The Bluetooth Special Interest Group (SIG), led by a nine-company Promoter group including 3Com, Ericsson, IBM, Intel, Lucent Technologies, Microsoft, Motorola, Nokia and Toshiba, is driving development of the technology and bringing it to market. For more information, see [4].

Wireless ad-hoc networks have been examined in some papers. The research has mostly been focused on routing (see, for example, Johnson [21]). Perkins [22] examined how mobile-IP could be used in ad-hoc networks. Davies *et al.* [23] suggested that token passing should be used in the MAC protocols for wireless LANs. However, only a few papers have examined Bluetooth. Haartsen [5] gives a good introduction to the technology used in Bluetooth (see also the Bluetooth Specification ver. 1.0 [6]). Johansson *et al.* [1][2][3] presents and examines a number of different network usage cases for Bluetooth.

In order to achieve good network performance, ad-hoc networks must ensure reliable data transfer for the applications. Therefore, the data packets in Bluetooth are protected by an ARQ scheme on the link layer. However, it is also necessary to have good end-to-end congestion control. The obvious solution is to use the Transmission Control Protocol (TCP), that guarantees reliable, in-sequence delivery of packets (see, for example, Stevens [10]).

The congestion control in TCP uses a dynamic sliding window to control the rate at which packets are transmitted. The congestion control scheme in TCP was first developed by Jacobson [9]. The most used version right now is called TCP Reno (see, for

example, Allman *et al.* [8]). Brakmo and Peterson [7] suggested a new congestion control for TCP, called TCP Vegas. Further, they showed that with TCP Vegas, the performance becomes better than with TCP Reno. This was also shown by Ait-Hellal and Altman [20] and Mo *et al.* [19]. Therefore, we will use TCP Vegas as the congestion control scheme in this paper.

One problem that is obvious when TCP is used in wireless networks is that TCP was originally designed for networks with low packet error rate. Wireless networks usually show high rate of packet losses and these losses may trigger the congestion control in TCP even though there is no congestion. Numerous papers have examined the performance of TCP over wireless links. Augé and Aspas [14], Lakshman and Madhow [18] and Gerla *et al.* [16] found that TCP does not work well in wireless environments. However, these papers assumed that there were no link layer retransmission schemes. Chaskar *et al.* [15] showed that a suitable link level error recovery mechanism may improve the TCP performance in wireless networks. Ludwig and Rathonyi [13] developed a link layer enhancement for GSM. Chandran *et al.* [17] suggested a feedback scheme for ad-hoc networks. Both Wong and Leung [11] and Chockalingam *et al.* [12] examined ARQ schemes for the link layer in wireless networks.

However, until now there has been no published papers on the performance of TCP over Bluetooth. In this paper, we present a detailed simulation model of TCP/IP over Bluetooth. We use this model to examine the maximum throughput and packet delays for TCP/IP traffic under various conditions. We find that Bluetooth is well suited to carry TCP/IP traffic and that both high throughput and low delays can be achieved also when the radio channel has a high packet error probability.

## 2. Bluetooth

Bluetooth is a short-range radio link which originally was developed to eliminate the cables between portable and/or fixed electronic devices. Today, Bluetooth is a true ad-hoc wireless network intended for both synchronous traffic and asynchronous traffic. Bluetooth consists of a number of protocols, all residing in the two lowest layers in the OSI-model, i.e. the physical and the data link layer. For a more detailed description of the system, the reader is referred to [6].

### 2.1 Baseband Protocol

In Bluetooth all units are peer units with identical hard and software interfaces. Two or more units sharing the same channel form a piconet, where one unit acts as a master (any unit may become a master), controlling traffic on the piconet, and the other units act as slaves.

The Bluetooth radio uses a fast frequency hopping scheme and short data packets to combat interference and fading. Furthermore, Bluetooth uses a slotted Time-Division Duplex (TDD) scheme for full duplex transmission, where each slot is 0.625 ms long.

The Bluetooth baseband protocol is a combination of circuit- and packet switching, hence, time slots may be reserved for both synchronous information (Synchronous Connection Oriented, SCO link) or dynamically allocated for asynchronous informa-

tion (Asynchronous ConnectionLess, ACL link). An SCO link will always be symmetrical, i.e. a down-link slot is followed by one uplink slot. An ACL link, however, may convey packets that covers several, continuous slots, either symmetric or asymmetric. In fact, there are two different ACL link packets, one denoted DM<sub>x</sub> for which the payload is FEC encoded and one denoted DH<sub>x</sub> for which the payload is unprotected (the packet header is always FEC encoded). The subscript *x* stand for the number of slots that is required to transmit the packet.

For ACL traffic, an unnumbered ARQ scheme is applied in which data transmitted in one slot is directly acknowledged by the recipient in the next slot. If the packet is not acknowledged a retransmission takes place immediately, and thus, delay due to errors is just one slot in duration.

The system offers a gross bitrate of 1 Mbps per piconet. Based on the TDD structure, Bluetooth supports an asymmetric ACL link of maximum 721 kbps in one direction while permitting 57.6 kbps in the other direction, or a symmetric ACL link of maximum 432.6 kbps. Several piconets may be linked together, thus forming a scatternet in which each piconet is identified by a unique hopping sequence, and as long as not too many, each provides a capacity of almost 1Mbps. Thus a scatternet consisting of say ten piconets, geographically on top of each other, may provide 10 Mbps.

## 2.2 Link Manager Protocol (LMP) and Logical Link Control and Adaptation Protocol (L2CAP)

LMP and L2CAP are layered above the Baseband Protocol and they reside in the data link layer. The LMP is used for link set-up and control and is not further considered in this paper. The L2CAP provides connection-oriented and connectionless data services to upper layer protocols with protocol multiplexing capability, packet segmentation and reassemble, and the conveying of quality of service information.

## 3. TCP Vegas

In this section we shortly describe the TCP congestion control scheme used in the investigations, the so called TCP Vegas. We describe the main parts of TCP Vegas. For more information about the scheme, please refer to Brakmo and Peterson [7].

We assume that a TCP connection has been set up between two nodes. Data segments are generated by an application. All segments are of the same size, called *Data Segment Size (DSS)*. TCP transmits the segments unchanged, i.e. no segmentation is performed on the TCP layer.

### 3.1 Round Trip Time Measurements

It is important that the control scheme has good estimates of the round trip time, *RTT*, for a segment, that is, the time from a segment is submitted until an acknowledgement, *ACK*, for the segment is received. When an *ACK* is received for a segment, that has not been retransmitted, the *RTT* for this segment is measured. A smoothed round trip time, *SRTT*, is then calculated as  $SRTT = SRTT + 0.125 \cdot (RTT - SRTT)$ .



This smoothed round trip time is used in the calculations of the retransmission timeout value,  $RTO$ , that determines when a segment should be retransmitted. The  $RTO$  is calculated as  $RTO = SRTT + 4D$  where  $D$  is the smoothed mean deviation of  $SRTT$ , given by  $D = D + 0.25 \cdot (|RTT - SRTT| - D)$ .

### 3.2 Estimation of Expected and Actual Rates

The *Expected* and *Actual* rates of a connection are used in the window adjustment algorithms below. The expected rate is the estimated transmission rate for a non-congested connection, whereas the actual rate is the estimated current transmission rate. These rates are estimated once every  $RTT$ , by using the  $RTT$  for a marked segment. During the  $RTT$  for this marked segment, the number of transmitted segments, *TransSeg*, are measured. The *Expected* rate is given by  $Expected = cwnd / BaseRTT$  where  $BaseRTT$  is the minimum  $RTT$  for all transmitted segments.  $cwnd$  is the current congestion window, which is used as an estimate of the number of segments in transit. The *Actual* rate is given by  $Actual = TransSeg / RTT$ .

### 3.3 Window Adjustments

The congestion control is basically a sliding window flow control. The control scheme is self-clocking, that uses *ACKs* from the receiver to submit new segments into the network. The sender adjusts the congestion window,  $cwnd$ , differently depending on the phase. When there are no *ACKs* to receive, as in the beginning of a transmission, the scheme is in the *slowstart* phase. When the transmission is running smoothly, the scheme changes to the *congestion avoidance* phase. After a segment loss, the scheme either enters a *fast retransmit and fast recovery* phase or returns to the *slowstart* phase.

Each time the window has been updated, the scheme checks if it is possible to transmit one or more segments. This is determined from the value of  $cwnd$  and the value of the advertised window in the receiver,  $rwnd$ . If the number of segments in transit is less than the minimum of  $cwnd$  and  $rwnd$ , new segments may be submitted.

**Slowstart.** The objective of the *slowstart* phase is to gradually increase the number of data in transit. Since there are no *ACKs* coming from the receiver, the scheme must be clocked in another way. First, the  $cwnd$  is set to one segment. The window is then increased by doubling the value of  $cwnd$  every other  $SRTT$ .

The scheme switches to the *congestion avoidance* phase when  $(Expected - Actual) < 1 / BaseRTT$ .

**Congestion Avoidance.** The objective of the *congestion avoidance* phase is to find a window size that allows a maximum throughput without causing congestion. Each time an *ACK* is received, the following is performed. If  $(Expected - Actual) < 1 / BaseRTT$  then  $cwnd$  is increased by  $1/cwnd$ , else if  $(Expected - Actual) > 3 / BaseRTT$  then  $cwnd$  is decreased by  $1/cwnd$ . Otherwise,  $cwnd$  is unchanged.

The scheme is in the *congestion avoidance* phase until a segment loss occurs. A loss is detected either by a retransmission timeout or by the arrival of a so called duplicated *ACK*. The receiver sends a duplicated *ACK* each time it receives a segment that is out

of order. When a loss is detected, the sender retransmits the lost segment and then changes to one of the other phases. If there is a retransmission timeout, the scheme switches to the slowstart phase. If the sender receives three duplicated ACKs for the same segment, it switches to the fast retransmit and fast recovery phase.

**Fast Retransmit and Fast Recovery.** This phase uses a parameter *sstresh* that is set to  $cwnd/2$ . The lost segment is retransmitted and then *cwnd* is set to  $sstresh+3$ . If the sender receives any more duplicated ACK for the retransmitted packet, *cwnd* is incremented by one for each ACK. When the first ACK arrives that acknowledges new data, *cwnd* is set to *sstresh*. After this the scheme switches to the congestion avoidance phase.

## 4. Simulation Model

We have developed a detailed simulation model containing both Bluetooth and TCP/IP. The network that is modelled is a piconet in Bluetooth with two nodes. The two nodes can for example be a laptop and a server.

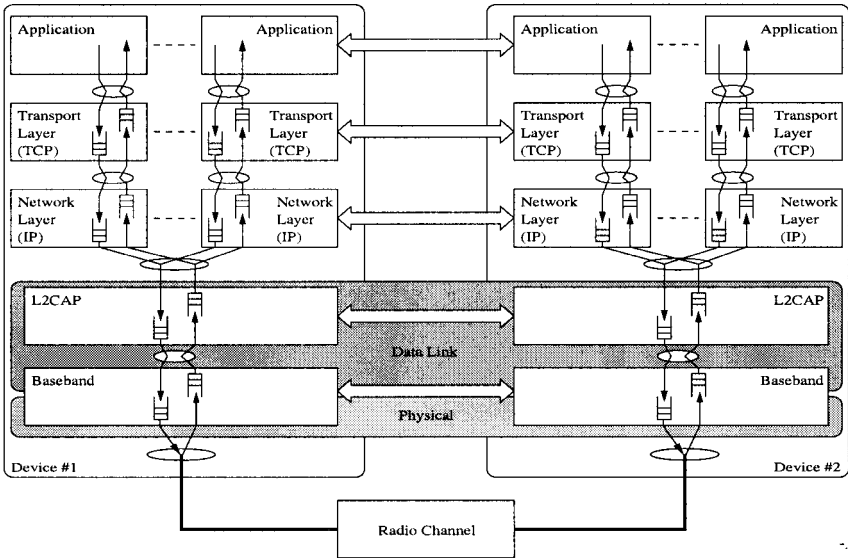
### 4.1 Transmission Procedures

The layers in the protocol stack are shown in Figure 1. The sender's application layer generates data segments according to some arrival process. The segments are directly transmitted to the TCP layer. The segments are put in the TCP sender buffer, where they are delayed at least  $D_{TCP}$  seconds. This delay models the processing needed to add a TCP header. When the window allows it, TCP sends the segments to the IP layer. At the IP layer, the segments are put in the IP sender buffer. At regular intervals of length  $D_{IP}$  seconds, the IP removes a segment from the buffer and sends it to L2CAP. The segments are put in the L2CAP sender buffer. L2CAP removes segments at regular intervals of length  $D_{L2CAP}$  seconds, divides the segment into Bluetooth packets and sends them to the Bluetooth Baseband sender buffer where they are delayed at least  $D_{Base}$  seconds. The Baseband transmits the packets according to the Bluetooth transmission scheme described earlier. Transmitted Baseband packets may be lost due to bit errors. The errors are modelled with a constant loss probability, in the following called Packet Error Probability (*PEP*).

When the receiver's Baseband layer receives a correct packet it is delayed  $D_{Base}$  seconds and sent to the L2CAP receiver buffer where it is delayed another  $D_{L2CAP}$  seconds. L2CAP reassembles the Bluetooth packets into TCP/IP segments and sends them to the IP receiver buffer. At regular intervals of length  $D_{IP}$  seconds, the IP layer removes a segment from the buffer and sends it to the TCP layer, where it is put in the TCP receiver buffer. When TCP sends an ACK for a segment, the segment is removed from the buffer and is sent to the application layer.

The ACKs are piggy-backed on TCP segments going in the opposite direction. According to TCP, the ACKs are only sent separately if these are delayed too much or if two or more TCP segments are waiting to be acknowledged.

We find it advisable to use long, uncoded packet types for data transmission since they have the largest ideal throughput. In [1], [2] and [3] the importance of multislot



**Figure 1.** Protocol layers and buffers in the simulation model

packets to increase throughput and keeping the delays low have been addressed. Uncoded packet types, i.e. DHx packets, have on average a 55% larger payload size than coded packet types, i.e. DMx packets. This means that 1.55 DMx packets are needed to be able to pass the same amount of information as can be passed in a single DHx packet. For a Packet Error Probability (PEP) of around 36% for DHx packets we obtain the same goodput as when using DMx packets not effected by errors. Further, the DMx packets can also become corrupted which means that even more DMx packets have to be sent. Therefore, we have decided to only use DHx packets in the simulations.

## 4.2 Data Packet Formats

A packet that arrives at the Bluetooth layer consists of three parts: A TCP header, an IP header and payload. L2CAP adds 4 bytes as channel identification and packet length. We have decided to use a total packet size that is as close to 1500 bytes (i.e the Ethernet packet size) as possible with the constraint that it should fit into a number of Bluetooth packets.

The TCP header is 32 bytes instead of the normal 20 bytes. The extra 12 bytes are necessary for the RTT calculations. The IP header is 20 bytes. With a payload of 1429 bytes, that is a total packet size of 1485 bytes, the packet fits into 55 Bluetooth packets of type DH1.

Normally, the ACKs are piggy-backed on TCP segments going in the opposite direction. When a separate ACK is sent this packet only consists of the headers, which means that it is 56 bytes long. A separate ACK is divided into 3 DH1 packets.

### 4.3 Model Parameters

The model parameters are shown in Table 1. The data segment size (DSS) is the one described above. The maximum receiver window is set as in [7]. The lower and upper bounds in the window adjustment algorithm ( $\alpha$  and  $\beta$ ) are set as in [7]. The other parameters are set to what we believe are realistic values for this system.

**Table 1.** Model parameters

Parameter	Value
Data segment size ( <i>DSS</i> )	1429 bytes
Maximum receiver window (Max <i>rwnd</i> )	12 segments, i.e 17.148 kbytes
Total buffer size on Bluetooth layer	15 kbytes
Segment delay on TCP layer ( $D_{TCP}$ )	1 $\mu$ s
Segment delay on IP layer ( $D_{IP}$ )	1 $\mu$ s
Segment delay on L2CAP layer ( $D_{L2CAP}$ )	1 ms
Segment delay on Baseband layer ( $D_{Base}$ )	1 ms
Lower bound in TCP Vegas window adjustment ( $\alpha$ )	1
Upper bound in TCP Vegas window adjustment ( $\beta$ )	3

## 5. Investigations

The objective of the investigations is to examine the combination of TCP/IP and Bluetooth.

### 5.1 Arrival Process

The arrival process determines how the application layer delivers data to the TCP layer. We use two different arrival processes. In the first case, we want to find the maximum throughput for the system, which means that the TCP layer must always have data to send. Therefore, the arrival rate of application data is always high enough in order for the queues on the TCP layer to be filled.

In the second case, we assume a more bursty application process. This process is modelled by an Interrupted Bernoulli Process (IBP), i.e. for a geometrically distributed period (active state) arrivals occur according to a Bernoulli process. This period is followed by another period (idle state) during which no arrivals occur, also this geometrically distributed. Given that we are in the active state, the process will stay in this state with probability  $1-p$  or it will go to the idle state with probability  $p$ . If the process is in the idle state it will stay there with probability  $1-q$  or go to the active state with probability  $q$ . When in active state, a slot contains a packet with probability  $\lambda=1$ . The time slots for the traffic generators are aligned with the time slots for the modelled piconet. The squared coefficient of variation,  $C^2$ , for the packet arrival times are used as a measure of burstiness in the simulations.

## 5.2 Simulations

Two types of simulations were performed. First, we examined the maximum throughput for a number of packet error probabilities (*PEPs*) under varying conditions. Second, we examined the delays, i.e. the time from a data segment is generated until it has been correctly received and the receiver has sent an ACK, for predetermined packet error probabilities.

As mentioned earlier Bluetooth supports an asymmetric link of maximum 721 kbps in one direction while permitting 57.6 kbps in the return direction, or a 432.6 kbps symmetric duplex link. In order to find the maximum throughput (goodput) over a disturbed Bluetooth channel, we generate data segments so that there is a segment to be sent whenever the window mechanism in TCP so allows.

The goodput figures obtained are used in the delay investigations as the "TCP channel capacity" which of course varies with different values of *PEP*. That means that when we in the delay investigations refer to a load on the channel this load is normalized to the TCP channel capacity at a certain *PEP*. If we generate data segments with a certain rate the load value will be different for different values of *PEP*.

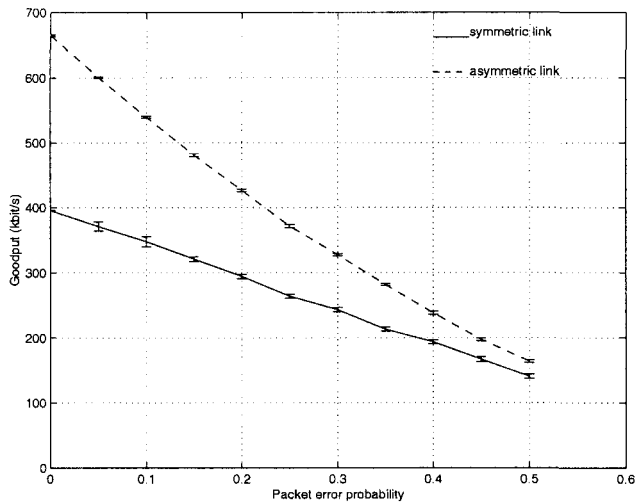
In the delay investigations we generate data segments according to the IBP described above. We have chosen two different values of the squared coefficient of variation,  $C^2$ , set to 1 and 50 respectively.

## 6. Results and Discussion

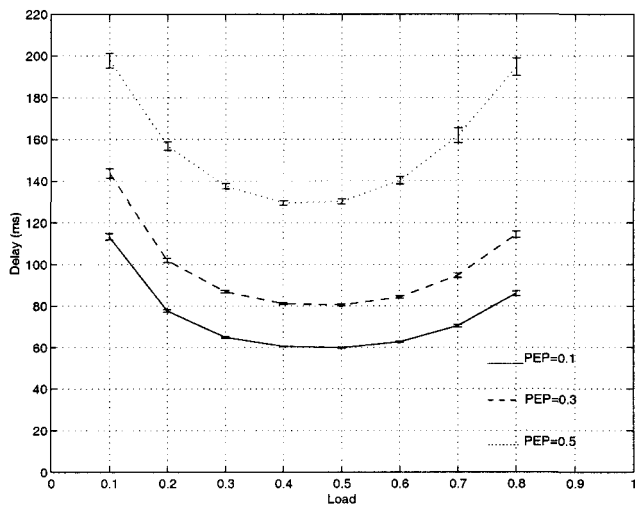
The two curves in Figure 2 show the goodput as a function of the packet error probability (*PEP*). The upper curve corresponds to an asymmetric case and the lower to a symmetric. We find the maximum goodput for *PEP*=0 to be slightly less than 721 kbps and 432.6 kbps respectively. The difference is due to headers and the TCP flow control mechanism. We see that the throughput decreases dramatically when the packet error probability grows. It is notable that the Bluetooth access scheme, based on polling, used for asynchronous traffic, can handle traffic also under very high loads. This is not the case for many other ad-hoc networks relying on contention based MAC principles.

In Figure 3 the delay when  $C^2=1$  is presented as a function of the normalized load. The three curves correspond to *PEP* equal to 0.1, 0.3 and 0.5 respectively. As we can see for low loads the delays decrease with load. This is due to that when the load is low the acknowledgment can just seldom be piggy-backed, and TCP then waits until there is a new "tick", which comes regularly every 500 ms, before it sends an acknowledgment. When the load increases, more and more acknowledgments are piggy-backed on TCP packets in the return direction. However, when the load is further increased the delay increases as packets are frequently queued up in different buffers along the path.

The squared coefficient of variation for the delays are shown in Figure 4. As load here is normalized it does not make sense to compare the three curves but more to see that  $C^2$  is quite low. We will be back to the shape of the curves when we discuss. Figure 6 also presents  $C^2$  as function of load but where we use another mechanism for sending non-piggy-backed acknowledgments.



**Figure 2.** Maximum throughput



**Figure 3.** Average packet delay when  $C^2=1$

Figure 5 shows the delays when data segments are generated with a squared coefficient of variation equal to 50. We see that the delays are dramatically higher compared to the case when the data segments were generated more smoothly, i.e. when  $C^2=1$ . However even in this case the squared coefficient of variation for the delays are of the order of one.

We have also elaborated with different sizes of various buffers. In each node the total buffer size for the Bluetooth layer, i.e. below the IP-layer, was set to 15 kbytes. We found no significant changes in neither throughput nor delay when these buffers were set to half the original sizes. Furthermore we have varied the lower and upper bounds in the TCP Vegas window adjustments, i.e. the values of  $\alpha$  and  $\beta$ . Also here, our per-

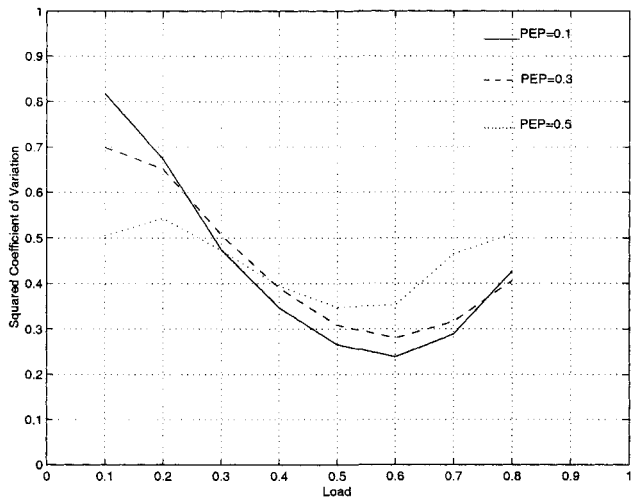


Figure 4. Squared coefficient of variation for the packet delay when  $C^2=1$

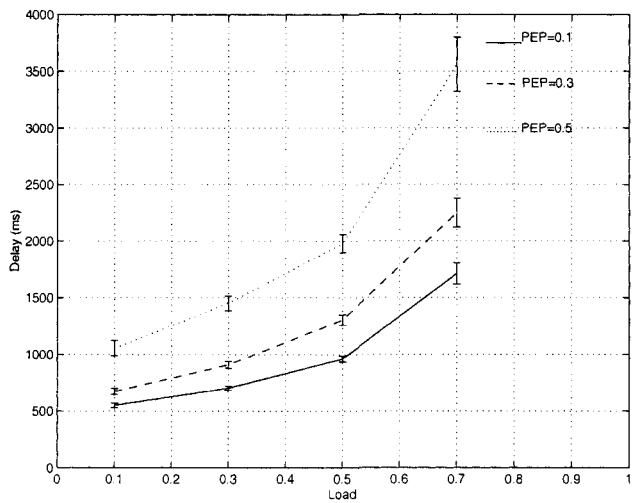


Figure 5. Average packet delay when  $C^2=50$

formance measures were not significantly affected by varying these parameters within reasonable values. Performance measures will most probably be more sensitive to the values of these two parameters, when more slaves are active in the piconet.

However, when we used a fix delay (200 ms) from the time a TCP packet is received until a non-piggy-backed acknowledgment is sent, the variances for the delays were significantly decreased, while the averages were hardly affected. Figure 6 shows the squared coefficient of variance as function of load for different packet error probabilities. The tendency we saw in Figure 4 is here further noticable. The normalized variations tend to grow for increasing low loads. For very low loads almost all TCP acknowledgments are handled by means of non-piggy-backed acknowledgements. As

load increases up to 0.2 or 0.3 more acknowledgements are brought piggy-backed and the mix of the two acknowledgement types leads to an increase in the delay variations. A further increase in load leads to that piggybacked acknowledgements do dominate and thus the variations in the delays due to the effects of a mix of different types of acknowledgements decreases. For high loads the delay variations naturally grows.

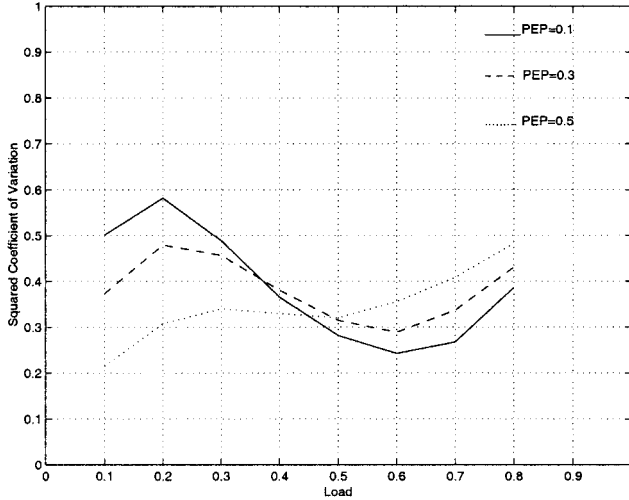


Figure 6. Squared coefficient of variation when  $C2=1$ .

## 7. Conclusions

Carrying TCP over wireless networks often leads to severe degradation in throughput and increased delays as the flow control mechanism in TCP reacts on delays introduced by retransmitting erroneous packets as if the network was congested. In this paper we have clearly demonstrated that the Bluetooth wireless ad-hoc network can handle TCP/IP traffic very well also under high packet error probabilities. Throughput is kept high and end to end delays are within reasonable limits also for quite high loads.

## References

- [1] P. Johansson, N. Johansson, U. Körner, J. Elg and G. Sennarp, "Short Range Radio Based Ad-hoc Networking: Performance and Properties", Proceedings of International Conference on Communications, Vancouver, Canada, June 1999.
- [2] N. Johansson, U. Körner and P. Johansson, "Wireless Ad-hoc Networking with Bluetooth", Proceedings of Personal Wireless Communication, Copenhagen, March 1999.
- [3] N. Johansson, U. Körner and P. Johansson, "Performance Evaluation of Scheduling Algorithms for Bluetooth", Accepted to IFIP Broadband Communications, Hong Kong, Nov 1999.
- [4] The Bluetooth Special Interest Group, Documentation available at <http://www.bluetooth.com/>



- [5] J. Haartsen, "*Bluetooth - The Universal Radio Interface for Ad-hoc, Wireless Connectivity*", Ericsson Review, No.3, 1998.
- [6] Specification of the Bluetooth System, ver. 1.0, July 1999.
- [7] L.S. Brakmo and L.L. Peterson, "*TCP Vegas: End to End Congestion Avoidance on a Global Internet*", IEEE Journal on Selected Areas in Communications, Vol.13, No. 8, Oct 1995.
- [8] M. Allman, V. Paxson and W. Stevens, "*TCP Congestion Control*", RFC 2581, April 1999.
- [9] V. Jacobson, "*Congestion Avoidance and Control*", ACM Sigcomm, Vol. 18, No.4, 1988.
- [10] W.R. Stevens, "*TCP/IP Illustrated, Volume 1: The Protocols*", Addison-Wesley, 1996.
- [11] J.W.K. Wong and V.C.M. Leung, "*Improving End-to-End Performance of TCP Using Link-Layer Retransmissions over Mobile Internetworks*", Proceedings of International Conference on Communications, Vancouver, Canada, June 1999.
- [12] A. Chockalingam, M. Zorzi and V. Tralli, "*Wireless TCP Performance with Link Layer FEC/ARQ*", Proceedings of International Conference on Communications, Vancouver, Canada, June 1999.
- [13] R. Ludwig and B. Rathonyi, "*Link Layer Enhancements for TCP/IP over GSM*", Proceedings of Infocom'99, New York, USA, March 1999.
- [14] A.C. Augé and J.P. Aspas, "*TCP/IP over Wireless Links: Performance Evaluation*", Proceedings of 48th Vehicular Technology Conference, May 1998.
- [15] H. Chaskar, T.V. Lakshman and U. Madhow, "*On the Design of Interfaces for TCP/IP over Wireless*", Proceedings of IEEE Military Communications Conference, Oct. 1996.
- [16] M. Gerla, R. Bagrodia, L. Zhang, K. Tang and L. Wang, "*TCP over Wireless Multi-hop Protocols: Simulation and Experiments*", Proceedings of International Conference on Communications, Vancouver, Canada, June 1999.
- [17] K. Chandran, S. Raghunathan, S. Venkatesan and R. Prakash, "*A Feedback Based Scheme for Improving TCP Performance in Ad-hoc Wireless Networks*", Proceedings of 18th International Conference on Distributed Computing, 1998.
- [18] T.V. Lakshman and U. Madhow, "*The Performance of TCP/IP for Networks with High Bandwidth-Delay Products and Random Loss*", IEEE/ACM Transactions on Networking, Vol.5, No.3, June 1997.
- [19] J. Mo, R.J. La, V. Anantharam and J. Walrand, "*Analysis and Comparison of TCP Reno and Vegas*", Proceedings of Infocom'99, New York, USA, March 1999.
- [20] O. Ait-Hellal and E. Altman, "*Analysis of TCP Vegas and TCP Reno*", Proceedings of International Conference on Communications, Montreal, Canada, June 1997.
- [21] D.B. Johnson, "*Routing in Ad Hoc Networks of Mobile Hosts*", Proceedings of IEEE Workshop on Mobile Computing and Applications, 1994.
- [22] C.E. Perkins, "*Mobile-IP, Ad-hoc Networking, and Nomadicity*", Proceedings of 20th International Computer Software and Applications Conference, 1996.
- [23] R.L. Davies, R.M. Watson, A. Munro and M.H. Barton, "*Ad-hoc Wireless Networking: Contention Free Multiple Access*", Proceedings of 5th IEE Conference on Telecommunications, Brighton, England, 1995.

# A Scheme for Time-Dependent Resource Reservation in QoS-Enabled IP Networks\*

Roberto Canonico, Simon Pietro Romano, Mauro Sellitto, and Giorgio Ventre

Dipartimento di Informatica e Sistemistica, Università di Napoli "Federico II",  
Napoli, Italy  
{canonico, sprom, sellitto, ventre}@grid.unina.it

**Abstract.** A number of distributed applications require communication services with Quality of Service (QoS) guarantees. The QoS provisioning issue in the Internet has been addressed by the IETF with the definition of the Integrated Services (IntServ) and Differentiated Services (Diffserv) frameworks. Resource reservation mechanisms on which these models are based are totally time-unaware. Yet, we believe that, in some cases, associating a time interval to network resource reservations could be useful for both users and network providers. In this paper we present a distributed scheme for time-dependent reservations in QoS-enabled IP networks. We also show how the standard signalling protocol RSVP may support this new reservation style, with only a few minor modifications. Finally, we present a first prototype implementation of the major component of the proposed architecture and we provide some hints on future applicability scenarios of the advance reservation paradigm and its impact on related topics such as policing and charging techniques in QoS-enabled IP networks.

## 1 Introduction

Existing communications systems are rapidly converging into an ubiquitous information infrastructure that does not distinguish between computing and communications, but rather provides a set of distributed services to users [1]. In this scenario the ability of the network to provide the applications with end-to-end Quality of Service (QoS) becomes a crucial issue. The best candidate infrastructure to support these new distributed services is the Internet, due to the enormous number of hosts already connected world-wide. Unfortunately, the communication service provided by current Internet does not offer QoS guarantees to applications. Having recognized the necessity of extending the basic best-effort service, the Internet Engineering Task Force has defined two architectures: *Integrated Services* [2] and *Differentiated Services* [3]. In particular, the former architecture enables applications to demand per-flow end-to-end guarantees from the network. To provide applications with the required QoS, all network routers along the path between sender and receiver must be kept informed about the characteristics of the flows, so to reserve the necessary

---

\* This work has been partially supported by the Ministero dell'Università e della Ricerca Scientifica e Tecnologica (MURST), Italy, in the framework of project MOSAICO

resources. A resource reservation protocol is thus needed to distribute this information over the network. For this purpose, the IETF has defined an appropriate signalling protocol, RSVP (*Resource reSerVation Protocol*) [4], whose role is to install a per-flow state in network routers, so to maintain a per-flow resource reservation along the communication path. To keep the connectionless approach of the Internet Protocol, RSVP mechanisms have been designed to continuously refresh resource reservations (*soft state*).

A major limitation of the IETF model is that it is totally time-unaware, i.e. reservations take place immediately and last for an unspecified duration. Many authors have already discussed the usefulness of an Advance Reservation Service [5, 6, 7, 8, 9]. In [6], for instance, Reinhardt focuses on the importance of this new kind of service for modern video conferencing and Video on Demand applications, whereas in [7] Wolf and Steinmetz also indicate manufacturing control systems and remote surgery as possible scenarios. In our opinion, the importance of time-dependent reservations in a QoS-capable Internet is of extreme importance because it allows a more efficient resource management in the network and can have a strong impact on pricing policies.

In this paper we compare some of the proposals that have emerged so far to support advance reservations in QoS-capable networks, and we present a new approach for Advance Reservations management, which is totally compatible with the Integrated Services Architecture. We also illustrate an implementation framework for our scheme, which relies on only a few minor changes to the RSVP protocol.

The rest of the paper is organized as follows. In Section 2 we discuss issues arising when time-dependent resource reservations are introduced in a QoS network. In Section 3 we present a scheme for distributed advance reservation in IP networks. In Section 4 the inner structure of the main component of our architecture is illustrated. Section 5 is devoted to the presentation of a prototype implementation of this component. Finally, Section 6 provides conclusions and discussion of future work.

## 2 Issues Related to Time-Dependent Reservations

A number of architectures have been already proposed to introduce Advance Reservation Services in a QoS network. A thorough description of them is presented in [7]. Some common architectural elements can be found in most of the proposed models. Both [8] and [9] propose a model relying upon a centralized approach. More precisely, they may be defined *server-based* as well as *routing-domain-based* since, for each Internet Autonomous System, they designate a host, named *Advance Reservation Server (ARS)* in [9] and *Advance Reservation Agent* in [8], whose purpose is to receive and manage all advance reservation requests. These works focus on the problems associated to the implementation of a *Reservation Management System*.

When adopting a server-based solution, an application may ask the ARS to set up an advance reservation and, after receiving a confirmation message, may terminate without any need to be active for the entire duration of the *bookahead* period. In this scenario, all state information relating to the advance reservation may be stored into a

stable memory (*hard-state*), thus eliminating all the problems associated with the existence of long-lived soft-state inside the routers.

A centralized solution has the advantage of relieving the routers from the burden of performing all the actions related to Admission Control. In fact, the introduction of an appropriate server relegates the routers to their normal forwarding and scheduling tasks. Nevertheless, the resulting architecture also shows the typical disadvantages of centralized systems: the creation of a performance and reliability bottleneck, poor scalability, and the need to keep in the centralized reservation server a consistent and up-to-date view of the present and future resource allocations throughout the network [5]. Furthermore, most of the signalling traffic associated to reservations would be concentrated on server nodes, causing them to become possible bottlenecks inside the network. A distributed solution, on the other side, shows a high degree of robustness with respect to crashes and uniformly allots control traffic across network nodes. The major drawback of such a solution remains in the augmented complexity of the routers.

All the proposed models for Advance Reservation Services rely upon a signalling protocol. For this purpose, an extension to RSVP seems both natural and simple to achieve. One could think of defining a new object used to specify *Time* as an additional QoS parameter. Such an object would be transferred across the network into RESV messages [10]. However, a number of issues arise, due to the specific mechanisms upon which the protocol relies. In particular, the main limitations deal with the receiver-oriented nature of RSVP and the soft-state paradigm that it adopts [6].

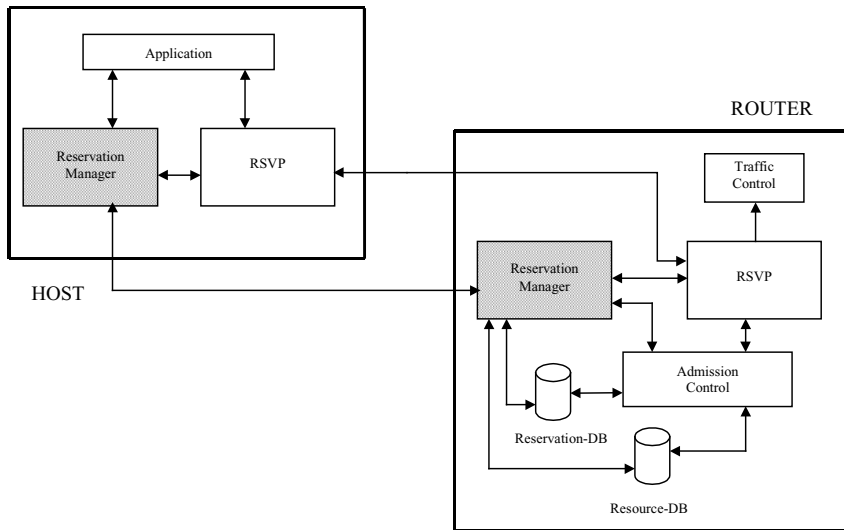
In the following we describe a distributed approach for management of time-dependent resource reservations which aims at introducing only a few modifications to the IETF Integrated Services Architecture. Our approach relies upon an augmented version of RSVP.

### 3 A Distributed Advance Reservation Scheme

Scope of this paper is to present a distributed solution for the implementation of a *Reservation Management System* [7], where the role played by RSVP is of crucial importance. It is our firm belief, in fact, that the advantages of such a solution largely overwhelm the minor drawbacks associated to router complexity.

In the architecture we propose, network elements (routers and hosts) assume the structure showed in Figure 1. The basic idea is to provide a *Reservation Manager* that takes up the responsibility of processing advance reservation requests during the negotiation phase, while relying on the well-known mechanisms of RSVP during the usage phase. Its main purpose is to perform admission control, *i.e.* to check whether enough bandwidth is available for the whole duration of the reservation interval. State associated to each accepted reservation is stored in a *Reservation-DB*, which also holds the information concerning bandwidth availability for each time slot. In this model, we cope with both immediate and advance reservations in a fashion that proves to be as less intrusive as possible with respect to existing RSVP-based QoS frameworks. Immediate reservations, in fact, are handled according to the standard *IntServ* model, with the only difference residing in the admission control module,

which is now in charge of taking into account the resources allocated both to immediate and advance reservations. However, the negotiation phase for an advance reservation is under the responsibility of *Reservation Managers*, which make use of both standard and *ad hoc* defined signalling messages.

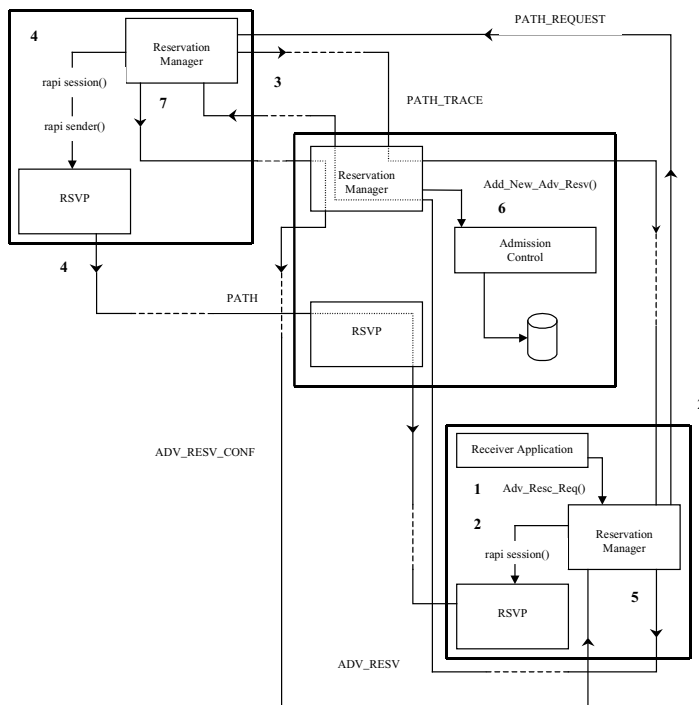


**Fig. 1.** Hosts and routers structure

Figure 2 depicts the steps associated with the negotiation of an advance reservation.

1. The receiving application sends the local *Reservation Manager* its reservation request.
2. To characterize the path between itself and the sender, the *Reservation Manager* of the receiving host executes a *rapi\_session()* call [11] to the local RSVP daemon. In order to tell its counterpart on the sending side to start sending path characterization messages, a new message is defined, called PATH\_REQUEST, which flows on the upstream direction from receiver to sender.
3. To appropriately trace the route between sender and receiver, the *Reservation Manager* on the sender side makes use of a PATH\_TRACE message containing an object similar to the standard RSVP\_HOP object. As it travels downstream, the PATH\_TRACE message is used to install path information in intermediate routers' *Reservation Managers*, needed to forward advance reservation messages in a further step.
4. Sender's *Reservation Manager*, based on the information contained into the PATH\_REQUEST message it received, executes both a *rapi\_session()* and a *rapi\_sender()* call, thus triggering the standard RSVP PATH sending phase. PATH messages flowing in the downstream direction are intercepted also by the *Reservation Managers* of intermediate nodes, which use them to build messages associated with advance reservations (ADV\_RESV messages).

5. Receiver's *Reservation Manager* notifies the receiving application of the arrival of PATH messages from the sender, thus allowing the application to make all the computations required to determine the parameters related to the desired QoS. The application then executes an appropriate *advance\_reserve()* call which triggers the creation and forwarding of an ADV\_RESV message from the receiver's *Reservation Manager* to the first router upstream.
6. Upon reception of the ADV\_RESV message, the router's *Reservation Manager* performs an admission test by calling an appropriate function (*Add\_New\_Adv\_Resv()*); if this test is passed, the ADV\_RESV message is further forwarded to the next upstream router, eventually reaching the sending host.
7. In case of positive response from all admission tests along the path, the receiver's *Reservation Manager* is notified with an ADV\_RESV\_CONF message, whose purpose is to confirm reservation acceptance; this in turn provides the application with a specific reservation identifier that will be used in the following phase associated with resource usage.



**Fig. 2.** The negotiation of an Advance Reservation

After the negotiation phase for an advance reservation, there is no need for the application that made the request to stay active even during the intermediate phase.

At the beginning of the usage phase, *Reservation Managers* automatically label the required resources as allotted to a pending reservation: data structures necessary to carry on classification and scheduling tasks are actually instantiated only when an explicit usage request from the involved application arrives. For this purpose, the receiving application uses a slightly modified version of the standard *rapi\_reserve()* call, containing a new parameter related to the identification of an already accepted and registered advance reservation. As a consequence, usage requests for advance reservations are easily handled as immediate reservations for which admission control tests consist in just verifying the existence, in the *Reservation-DB*, of an advance reservation with the same identifier as the one provided by the *rapi\_reserve()* call.

Our model is thus able to cope with both the major problems associated with the use of RSVP in an advance reservation framework, *i.e.* soft state and receiver-oriented approach. The issue related to the possible absence of the sender is, in fact, accommodated by the introduction of the *Reservation Manager* entity, whereas all problems concerning the soft state are eliminated thanks to the Reservation and Resource databases, which guarantee that, for the entire duration of the intermediate phase, no soft state has to be managed by network nodes.

## 4 Reservation Manager Mechanisms

The goal of the *Reservation Manager* is to cope with the absence, in RSVP, of a set of suitable mechanisms to handle reservations in advance. In the following we analyze in more detail some of the relevant characteristics of the overall framework.

### 4.1 Negotiation Handshaking

The main function of *Reservation Managers* is to process advance reservation requests. To accomplish this task, they exchange a number of messages needed to decide whether or not a specified request must be accepted and register state information inside the routers involved in the reservation process. Messages exchanged during this phase are: PATH\_REQUEST, PATH, PATH\_TRACE and ADV\_RESV, as showed in Figure 3.

It is worth noting the introduction of a PATH\_REQUEST message, to deal with the possible absence of the sender at the time when an advance reservation request is made from a potential receiving application. With this new message the receiving application may solicit the *Reservation Manager* at the sender side to start sending PATH messages, thus acting as if the actual sender were present. These PATH messages, however, are only needed to gather all the information related to the path from sender to receiver (contained, for example, in the ADSPEC object). The specific format of this and other new messages introduced is detailed in [12].

Faults, as well as re-routings, occurring during the intermediate phase between negotiation and usage should be taken into account by the *Reservation Managers* with the help of *Resource Managers*; nevertheless, these issues are beyond the scope of a trial implementation and we left it for further development of the overall framework.

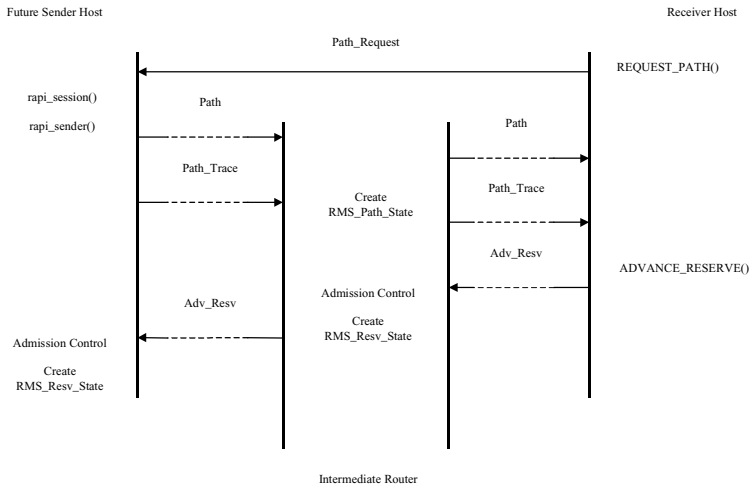


Fig. 3. Typical handshaking for the negotiation phase of an Advance Reservation

## 4.2 Confirmation of Reservation Acceptance

Each node, upon reception of an `ADV_RESV` message, has to submit the request to the Admission Control module. If the test is passed successfully, a local reservation identifier is returned and a `RMS_Resv_State` object is built with the purpose of storing all the information related to the advance reservation. This object is inserted into the reservation database only when admission control has been successful on each and every node along the path. This requires the introduction of a new message, especially tailored to accomplish this task. This new message, called `ADV_RESV_CONF`, is showed in the lower part of Figure 4.

## 4.3 Flowspecs Merging

Flowspecs merging in the case of advance reservations introduces a new dimension with respect to immediate reservations. No merging is possible if the time intervals of two different reservations do not intersect. Things change when at least a partial overlapping exists, as depicted in Figure 5.

A simple analysis of Figure 5 shows that merging is allowed if and only if the new request arriving at a node is “fully included” into a reservation already in place, *i.e.* if its *flowspec* turns out to be “less than or equal to” [4] the already existing one and the reservation interval for the latter request is included into the time interval of the former. The admission test for a new request, however, must always check whether the interested flow had already been assigned a portion of resources, even in a sub-interval of the one conveyed inside the request. For instance, when `RESV3` arrives (Figure 5), the bandwidth considered for admission test must be `B`, instead of `6B`, since the flow has already been assigned a bandwidth of `5B` with `RESV1`.



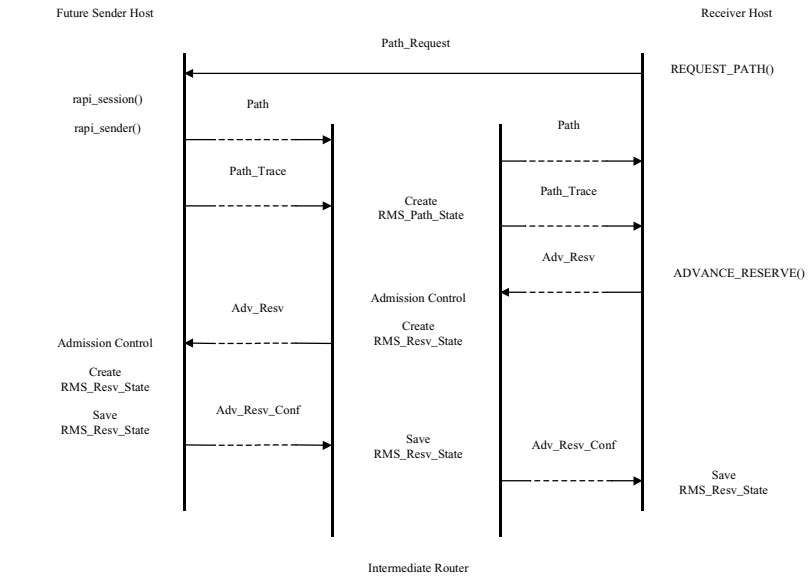


Fig. 4. Confirmation message for advance reservations

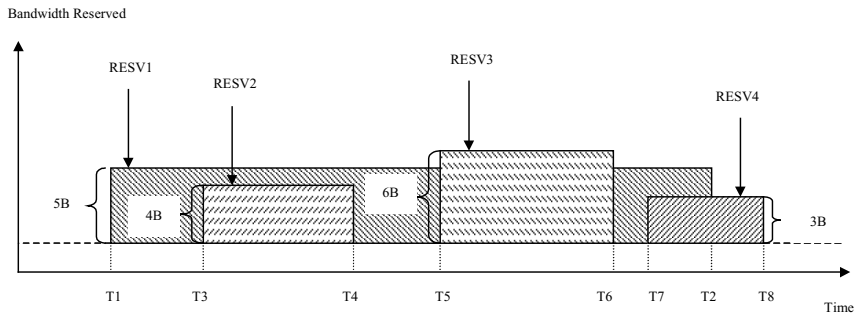
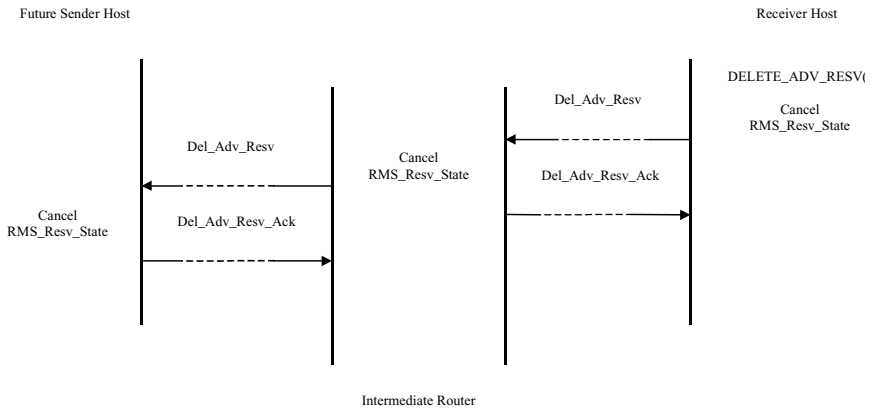


Fig. 5. Flowspec merging in the case of advance reservation

4.4 Reservation Cancellation

An advance reservation may be cancelled from the requesting application via the `delete_adv_resv()` function, which causes all the state information stored into the receiver’s database to be deleted. To achieve the same goal on all the nodes along the path, the *Reservation Manager* at the receiving side sends upstream a `DEL_ADV_RESV` message that triggers the cancellation process on each intermediate element while crossing the network. The cancellation process becomes a little tricky in the presence of merged reservations. A thorough discussion of the issues involved in this case may be found in [12]. Furthermore, it should be

considered the case of a lost DEL\_ADV\_RESV, resulting in persistence of state information associated to a deleted reservation into some of the intermediate nodes' databases. To solve this problem, an acknowledgement message, called DEL\_ADV\_RESV\_ACK, has been introduced, with the purpose of confirming the cancellation of a specified reservation. The reservation cancellation process is showed in Figure 6.



**Fig. 6.** Reservation cancellation

#### 4.5 Error Handling

An advance reservation request which fails the admission control test on a node, causes an error message, called `ADV_RESV_ERR`, to be generated and sent downstream towards the receiving host. Each intermediate node that gets this message deletes all state information concerning the request, whereas the receiving host also uses an upcall routine to inform the receiving application about the error. The same technique is used to deal with other errors, such as those relating to the absence of a suitable `RMS_Path` State object for a session, as well as to the presence of an unknown or conflicting reservation style.

#### 4.6 QoS Enforcement Phase

QoS enforcement for advance reservation does not differ from the setup of an immediate reservation. As already mentioned, this new kind of reservations may be seen, during this phase, as usual immediate reservations for which the admission control test simply consists in a check into the *Reservation-DB* aiming at verifying the existence of a `RMS_Resv` State object matching the parameters contained in the usage request. This object includes information concerning the time interval for which the reservation is being requested and a handle for reservation authentication inside the node database. These new parameters must thus be included in the usage request too. This is accomplished by exploiting the RSVP standard interface by means of the `reserve()` function, slightly modified to the purpose. The execution of such primitive

causes modified (*i.e.* including time interval and reservation handle) RESV messages to be generated at the receiving application’s side. Upon reception of this modified RESV message, each node is able to realize whether the request is for an immediate or an advance reservation and may consistently choose the right admission control algorithm to be performed.

### 5 A Prototype Reservation Manager

The *Reservation Manager* is responsible for the management of time-dependent reservations in the overall system. Its main role is admission control. State associated to each accepted reservation is stored in a *Reservation-DB*, which also holds the information concerning bandwidth availability for each time slot. The system is also responsible for monitoring reservation state changes, with special regard to transitions from established to pending states, and from pending to expired state [7]. The main components of the Reservation Manager are showed in Figure 7.

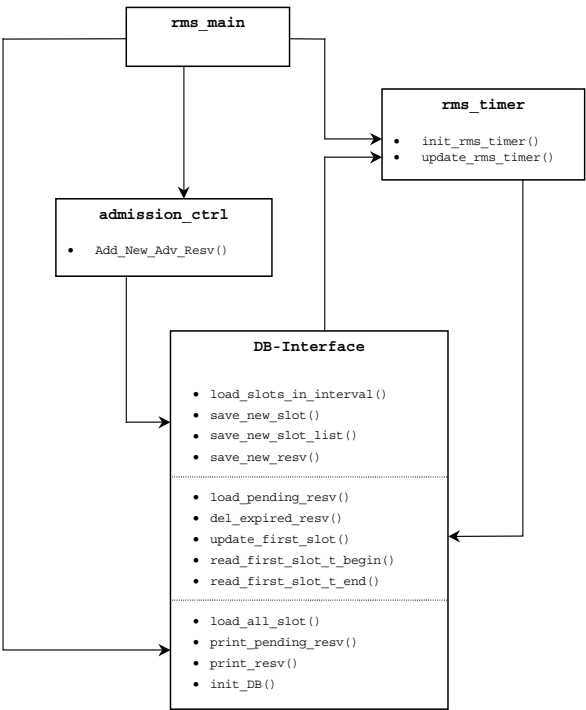


Fig. 7. Structure of the Reservation Manager

A time-dependent reservation may be required via the user interface provided by the `rms_main` module. Following a reservation request, the `admission_ctrl` module is activated to perform the admission test for that flow. To the purpose, it interacts with the *Reservation-DB*, both to retrieve information concerning bandwidth availability in the specified time interval and to store data structures associated to new reservations. The last module showed in the figure relates to timer management. It manages a sorted time slots list. An appropriate timer is set so to expire as soon as the time slot on top of the list starts, thus triggering the activation of a handler. This function is in charge of doing all the actions associated to the transitions from established to pending states (for reservations waiting to be activated) and from pending to expired (for reservations which were supposed to be activated during the previous slot).

A prototype of a Reservation Manager compliant with the model presented in Figure 7 has been implemented for the FreeBSD operating system. This module interacts with a traffic control module based on a WFQ packet scheduler [13].

## 6 Conclusions and Future Work

In this paper we have presented a scheme for distributed advance reservation in QoS-enabled IP networks. Our original contribution consists in the design of a distributed solution both efficient and simple to implement. As for applicability is concerned, we went into the details of a solution especially suited to extend the IETF Integrated Services model. Since we are firmly convinced that, while taking into account user expectations and needs, it is compulsory to consider Quality of Service as an end-to-end issue, we assumed the IntServ model as a starting point. In our framework, we are now finished with the development of a prototype Reservation Manager and work is currently in full swing to integrate this new model inside current RSVP protocol implementations.

Work is in progress to design a model for Virtual Private Networks comprising both IntServ and Diffserv components and aiming at the definition of a general environment, exploiting advanced policing and charging mechanisms to provide QoS at different levels of granularity. More specifically, we envision the use of our signalling protocol both at a microflow level, as discussed in this paper, and at a higher hierarchical level, as a means for implementing the handshaking required between Bandwidth Brokers [14] for dynamic negotiation of Service Level Agreements. In this scenario the network is considered as an entity which is able to provide a service general enough to include both immediate and advance reservations while leveraging on charging to enforce differentiation according to specific service parameters.

## References

1. D. Clark, J. Pasquale *et al.*. "Strategic Directions in Networks and Telecommunications". ACM Computing Surveys, No. 4, Dec. 1996, pp. 679-690.

2. R. Braden, D. Clark, and S. Shenker. "Integrated Services in the Internet Architecture: an Overview". *RFC 1633*, July 1994.
3. S. Blake, D. Black, M. Carlson, E. Davies, Z. Wang, and W. Weiss. "An Architecture for Differentiated Services". *RFC 2475*, Dec. 1998.
4. R. Braden, L. Zhang, S. Berson, S. Herzog, and S. Jamin. "Resource ReSerVation Protocol (RSVP) -- Version 1 Functional Specification". *RFC 2205*, September 1997.
5. D. Ferrari, A. Gupta, and G. Ventre. "Distributed Advance Reservation of Real-Time Connections." In Proc. of the Fifth Int. Workshop on Network and Operating System Support for Digital Audio and Video, Durham, NH, USA, April 1995.
6. W. Reinhardt. "Advance Resource Reservation and its Impact on Reservation Protocols". Proceedings of Broadband Island '95, Dublin, Ireland, September 1995.
7. L. C. Wolf, R. Steinmetz. "Concepts of Resource Reservation in Advance". Special Issue of Multimedia Tools and Applications on "The State of The Art in Multimedia", May 1997 (Vol. 4, No. 3).
8. O. Schelén, S. Pink. "An agent-based architecture for advance reservations". In IEEE 22nd Annual Conference on Computer Networks, Minneapolis, Minnesota, November 1997.
9. S. Berson, R. Lindell, R. Braden. "An Architecture for Advance Reservations in the Internet". USC Information Sciences Institute, July 16, 1998.
10. J. Wroklawsky. "The use of RSVP with IETF Integrated Services". *RFC 2210*, September 1997.
11. R. Braden and D. Hoffman. "RAPI -- An RSVP Application Programming Interface - Version 5". IETF Internet Draft draft-ietf-rsvp-rapi-01.txt, Aug. 1998.
12. M. Sellitto. "Uno schema per la riservazione time-dependent di risorse in reti IP a Qualità di Servizio". (In Italian). M.Sc. Thesis, University of Naples, July 1999.
13. R.D'Albenzio, S.P. Romano and G. Ventre. "An Engineering Approach to QoS Provisioning over the Internet". Lecture Notes in Computer Science no. 1629, Springer, May 1999, pp. 229-245.
14. K. Nichols, V. Jacobson, and L. Zhang. "A Two-bit Differentiated Services Architecture for the Internet". Internet draft, draft-nichols-diff-svc-arch-00.txt, Nov. 1997.

# Impact of “Trunk Reservation” on Elastic Flow Routing

Sara Oueslati-Boulahia and James W. Roberts

France Telecom-CNET

38-40 rue du Général Leclerc, 92794 Issy-les-Moulineaux Cedex 9, France  
{sara.boulahia,james.roberts}@cnet.francetelecom.fr

**Abstract.** This paper focuses on routing strategies for elastic flows within a flow-based routing architecture. In contrast to other elastic flow routing studies, we assume that elastic flows have a minimum guaranteed rate and are, therefore, subject to admission control. Our ultimate goal is to devise an adaptive algorithm that maximizes the throughput of elastic flows at light load and that preserves network efficiency under overload conditions. To this end, we examine in this paper the impact of trunk reservation on elastic flow quality of service. Trunk reservation is a technique commonly used in circuit-switched networks to prevent performance degradation under overload conditions. Two algorithms integrating a form of trunk reservation are proposed and their performance compared by means of simulation to the performance achieved by an alternative algorithm that we proposed and evaluated in an earlier study [1]. Our results highlight interesting features of trunk reservation when used for elastic flow routing.

## 1 Introduction

Several studies in the past have considered the problem of routing flows with bandwidth or delay guarantees [2–4]. Recently, there has been a growing interest in addressing issues related to the routing of best effort flows in multiservice networks [1, 5–9]. This paper considers a microflow-based routing architecture for Internet-like networks. More precisely, we assume routing is performed for individual transactions, each transaction being represented as a continuous flow of packets. In practice such a flow might be identified by the five tuple ( *source address*, *destination address*, *source port number*, *destination port number*, *protocol identifier* ) together with the fact that successive packets are not separated by more than a certain time interval. We do not consider here the feasibility of routing objects defined at such a fine granularity which is clearly an important issue. We seek rather to evaluate the potential advantage of flow sensitive routing algorithms with respect to quality of service and network traffic handling efficiency.

The service model we consider in our routing architecture was proposed in [10] and distinguishes two classes of flow, elastic flows and stream flows. Elastic flows result from the transfer of a digital object (text, image, video) from one place

to another. They are characterized by a volume (in bytes) and their transfer rate adjusts to the amount of available bandwidth as a result of being controlled by a mechanism such as TCP. Their transfer time depends on the throughput they achieve which thus constitutes their main QoS parameter. Note also that the faster elastic flows complete, the sooner resources are released for ongoing and future flows. Consequently, enhancing elastic flow throughput positively impacts both elastic flow QoS and network resource utilization. Stream flows, on the other hand, mainly result from audio and video applications. Unlike elastic flows, stream flows have an intrinsic rate (possibly variable) and an intrinsic duration and generally correspond to delay sensitive applications.

We assume here that elastic flows are guaranteed a minimum rate by means of admission control. In other words, new flows will not be admitted if the rate of ongoing elastic flows would consequently be reduced below a certain limit. This additional feature impacts the design of elastic flow routing algorithms. The guarantee of a minimum rate to elastic traffic is largely motivated in [5, 6]. The detailed admission control procedure envisaged for a multiservice network integrating elastic and stream traffic, as well as the underlying resource sharing model, are described in [1].

The present study focuses on routing strategies for elastic flows. Our objective is to devise a routing algorithm that enhances as much as possible the throughput of elastic flows under light traffic conditions, while efficiently handling traffic at overload. Initial results obtained in [7] suggest that, in order to maximize the throughput of best effort flows, least loaded paths should be preferred to minimum-hop paths under light load conditions, and that shortest paths should, on the contrary, be privileged under heavy load conditions. Ideally, the path selection algorithm does this automatically without requiring a manually specified threshold that distinguishes between heavy load and light load. In this perspective, we proposed and evaluated in earlier work a novel routing algorithm, called *Maximum Utility Path*, which proved to be particularly robust with respect to topology configurations and traffic conditions [1, 11].

In this paper, we investigate for the first time the benefit for elastic flow routing of the technique widely used in circuit-switched networks to prevent performance degradation under overload conditions known as *trunk reservation*. More precisely, we evaluate the performance of two algorithms that integrate the trunk reservation technique. The latter algorithms derive from two well known algorithms, *Widest-Shortest Path* and *Shortest-Widest Path*. The performance of the proposed algorithms, called *Widest-Shortest Path with Trunk Reservation* and *Shortest-Widest Path with Trunk Reservation*, are compared to that obtained with the *Maximum Utility Path* algorithm, under various load conditions and on different network topologies. In all our simulation scenarios, only elastic traffic is offered to the network.

The next section discusses the benefit of trunk reservation for enhancing elastic flow QoS under heavy traffic conditions. In Section 3, we describe our routing framework: we present the link metrics we consider, discuss admission control issues and describe the routing algorithms to be evaluated. The following

section presents our simulation model and in Section 5 simulation results are analyzed. We conclude the paper by a summary of our main simulation results.

## 2 Trunk Reservation for Elastic Flows

Trunk reservation is an easily implemented control mechanism that is used in circuit-switched networks to protect the network against instability due to the excessive use of alternative paths during overload conditions. An incoming call that is rejected by its direct path is admitted on an alternative path, with two or more hops, only if the number of free circuits on all the links of the path is more than a certain threshold. The implementation of trunk reservation in telephone networks ensures that direct calls are given priority over less efficient overflow calls. One potential concern with the trunk reservation scheme is how to choose the threshold value. Instead of an absolute threshold we use the notion of *Trunk Reservation Level* (TRL) which represents the proportion of capacity that is reserved to priority flows. Several studies [12–14] have attempted to identify optimal values for the TRL which typically represents a small proportion of accessible trunks.

In this paper, we focus on networks carrying elastic flows with a guaranteed minimum rate. In this context, by trunk reservation we mean the reservation of a certain proportion of bandwidth for minimum-hop path routed traffic. The amount of bandwidth that should be reserved is one object of our investigation. Note that in the context of elastic flow routing, the throughput of flows constitutes an additional performance metric to be taken into consideration. In the remainder of this paper *directly-routed traffic* refers to flows routed on their minimum-hop path(s) (not necessarily direct paths) and *overflow traffic* refers to flows rejected by their minimum-hop path(s).

At this point we investigate the benefit of trunk reservation for elastic flows at the level of a single link. Consider a router outgoing link of capacity  $C$  simultaneously used by  $N$  elastic flows belonging to two different classes : directly-routed traffic (class 1) and overflow traffic (class 2). The difference resides in the access priority. Flows of the second class have a more restricted access than those of the first class. The link is modeled by a Processor Sharing queue with two thresholds  $S_1$  and  $S_2$  with  $S_1 \geq S_2$ . This implies that when  $N$  flows are in progress, each receives a throughput equal to  $\frac{C}{N}$ , idealizing the fair sharing objectives of TCP congestion avoidance algorithms. A flow of class  $i$  is admitted only if the number of ongoing flows is smaller than  $S_i$ ,  $i = 1, 2$ . We take  $C = 100$  and set the minimum rate for all flows to 1% of  $C$ , so that  $S_1 = 100$ . The relation between  $S_1$ ,  $S_2$  and the TRL value is given by:  $S_2 = S_1 \times (1 - TRL)$ .

We further assume a Poisson arrival process of intensity  $\lambda_i$  for flows in classes  $i$ ,  $i = 1, 2$ . The size of flows in both classes has a general distribution with mean  $\frac{C}{\mu}$ . Let  $B_i$  denote the blocking probability for class  $i$ .  $B_1 = P[N \geq S_1]$  and  $B_2 = P[N \geq S_2]$ . Note that the mean sojourn time is the same for both classes, because, once in the system, all flows get equal link bandwidth shares.  $B_1$  and  $B_2$  are given by the following expressions:



$$B_1 = \rho^{S_2} * \rho_1^{S_1 - S_2} * P[N = 0] \quad (1)$$

$$B_2 = \frac{\rho^{S_2} * (1 - \rho_1^{S_1 - S_2 + 1}) * P[N = 0]}{1 - \rho_1} \quad (2)$$

with

$$P[N = 0] = \left( 1 + \frac{\rho * (1 - \rho^{S_2})}{1 - \rho} + \frac{\rho_1 * \rho^{S_2} * (1 - \rho_1^{S_1 - S_2})}{1 - \rho_1} \right)^{-1}, \quad \rho_1 = \frac{\lambda_1}{\mu}, \quad \rho = \frac{\lambda_1 + \lambda_2}{\mu}$$

For offered loads less than 1, both  $B_1$  and  $B_2$  are very small for any reasonable choice of  $S_2$  (e.g.  $S_2 \geq 40$ ). Differentiation is significant mainly in overload conditions ( $\rho > 1$ ). Figures 1 and 2 plot the blocking probability of each class of flow for  $\rho = 1.2$  and  $\rho = 2.4$ , respectively. Flows in both classes have the same arrival intensity ( $\lambda_1 = \lambda_2$ ).

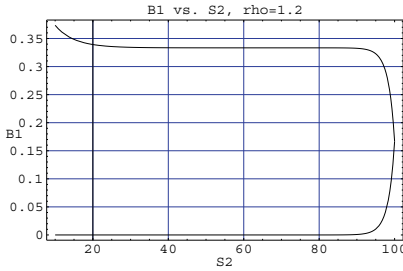


Fig. 1.  $\rho = 0.6 * 2 = 1.2$

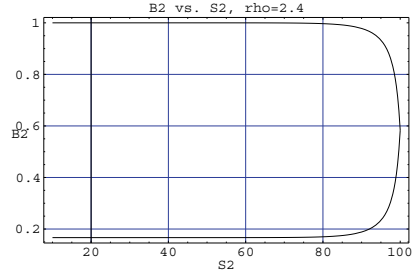


Fig. 2.  $\rho = 1.2 * 2 = 2.4$

We observe that offering a limited access to overflow traffic on a link effectively protects flows for which this link composes their minimum-hop path. For a load of 1.2, only overflow traffic is blocked and when the load is high enough to cause blocking to class 1 flows, virtually all class 2 flows are blocked. We also note that this protection is effective for a wide range of threshold values  $S_2$ . Blocking probabilities of both classes converge when  $S_2$  approaches the value of  $S_1$ .

The use of trunk reservation on any single link of a network has ramifications which may extend to the rest the network. Our objective in the remainder of the paper is to examine the impact of trunk reservation on elastic flow performance at the level of a network. To this end, we consider two algorithms incorporating trunk reservation and compare their performance to that provided by the algorithms from which they derive.

### 3 Routing Framework

In this section, we describe the link metrics we consider, specify the condition for admitting a new flow and present the routing algorithms to be evaluated.

### 3.1 Link Metrics and Admission Conditions

In our routing architecture, elastic flows are assured a common minimum rate. The imposition of this minimum rate is intended to avoid the negative impact of very low rates occurring notably in situations of overload and should not be viewed as a customizable service guarantee. Users need not know about it at all.

All the elastic flow routing algorithms described in the following section use the *offered rate* as a link metric. The offered rate represents an equal fair share of the link bandwidth available to elastic flows. In the present study, the link capacity  $C$  is entirely dedicated to elastic traffic and the offered rate is given by:  $r = C/(N_e + 1)$ . Assuming a fair bandwidth sharing flow control mechanism, the fair share rate offered by a given link is tightly correlated to elastic flow QoS. The offered rate on a link actually represents a lower bound on what flows can actually attain due to the rate limitations affecting competing flows on other links. Of course, the new flow will not realize this rate if its bottleneck link is elsewhere. The path offered rate is equal to the smallest offered rate of all the links constituting the path. A path is said to be feasible for an elastic flow if its offered rate is at least equal to the minimum rate.

Note that maintaining a count of ongoing elastic flows assumes that nodes are able to detect a new flow and determine if a flow is still active or has expired. Besides, in a link state approach the metric *current elastic flow count* would need to be advertised using a certain update mechanism. The potential inaccuracy induced by the update mechanism and its impact on routing performance are not considered in this paper. In addition to this dynamic metric, a static route metric representing the number of hops along a route is used to allow resource usage control.

### 3.2 Description of the Routing Algorithms

We evaluate two novel routing algorithms that integrate a form of trunk reservation: *Widest-Shortest Path with Trunk Reservation* (WS/TR) and *Shortest-Widest Path with Trunk Reservation* (SW/TR). Both algorithms are meant to enhance the performance of the algorithms *Widest-Shortest Path* (WS) and *Shortest-Widest Path* (SW), respectively. An alternative algorithm, called *Maximum Utility Path* (MU) introduced in [1], is also presented with the objective of comparing its performance to that achieved by the two new algorithms.

A number of studies [1,7,11] have evaluated the performance of WS and SW for elastic flows. WS selects the feasible path with the largest offered rate among those with the smallest number of hops. This algorithm privileges "cheap" paths to more resource consuming paths with higher offered rates, and performs therefore better at high load than at light load. One concern with this algorithm is its sensitivity to topology configurations which may exacerbate its poor performance under light load conditions by limiting its ability to perform load balancing (due to the absence of several paths which are equivalent in terms of number of hops). SW, on the contrary, selects the path offering the highest

rate with the number of hops deciding ties. Results obtained in [1, 7] show that SW yields high per-flow throughput at light load, but its performance rapidly deteriorates as the load increases due to excessive resource consumption.

A possible enhancement to WS and SW is to use “trunk reservation” to prevent the use of long paths when the network is congested. WS/TR (resp. SW/TR) performs like WS (resp. SW) except that non minimum-hop paths are feasible only if a certain proportion of bandwidth remains available on their bottleneck link to “directly-routed” flows.

MU, on the other hand, evaluates the utility of routing a flow on each path and chooses the path with maximum utility. The following utility function is defined:

$$U(P, B) = \log(r) - B * \Delta h \quad (3)$$

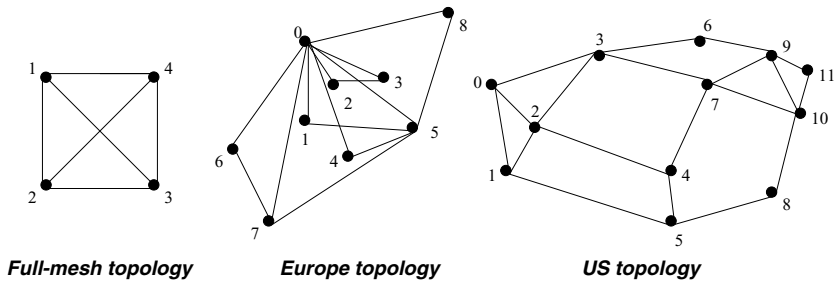
where  $r$  represents the offered rate along path  $P$ ,  $\Delta h$  represents the difference in the number of hops between path  $P$  and the shortest feasible path, and  $B$  is a constant (set to 1 in this study). The utility of a route increases with  $r$  and decreases linearly with the number of extra hops  $\Delta h$ . An interesting property of this algorithm is its overflow condition, i.e., the condition for selecting a route longer than the shortest feasible one. Consider two feasible candidate paths  $P_0$  and  $P_1$  with  $h_0$  and  $h_1$  hops, respectively. Assume  $P_0$  is the shortest feasible route. Let  $r_0$  and  $r_1$  denote the respective bottleneck rate for each route. The algorithm will select  $P_1$  instead of  $P_0$  if:  $\frac{r_1}{r_0} > e^{B*(h_1-h_0)}$ . From this condition, it appears that the probability of selecting a path longer than the shortest feasible path decreases exponentially with the number of extra hops on that path. Note that by varying  $B$  from 0 to  $\infty$ , we cover the spectrum between the *Widest Path* algorithm (with  $B = 0$ ), and the *Minimum-Hop Path* algorithm (with  $B = \infty$ ). Another interesting property of this algorithm is that it automatically places more emphasis on resource conservation as load increases due to the strict concavity of the log function.

## 4 Simulation Model

Our evaluation is based on the use of an event-driven simulation that performs path selection, admission control and bandwidth reservation operations at the flow level. In the simulations, an elastic flow has an instantaneous throughput equal to the bottleneck offered rate of its route. This choice is less efficient than max-min fair bandwidth sharing but preserves its essential properties while reducing simulation complexity.

Figure 3 illustrates the network topologies considered in this study. The Europe and US topologies are derived from actual commercial backbone networks. The topologies considered exhibit different characteristics with regard to size, the degree of connectivity (the US topology is loosely meshed, the Europe topology is rather tightly meshed and the four-node topology is completely meshed) and symmetry.

The traffic model considers elastic traffic only. We assume a Poisson arrival process for elastic flows with random selection of the source-destination pair



**Fig. 3.** Network topologies considered

according to equal probabilities (uniform traffic distribution). Elastic flows are characterized by a minimum rate equal to 1 BU, and an access rate which may limit their throughput. We further assume that elastic flow size follows a Pareto distribution with parameter  $\alpha = 1.5$ . With the Pareto parameters we consider, the resulting average size of flows is 0.66 BU second, and 90% of elastic flows have a size less than or equal to 1 BU second.

Elastic traffic performance is measured jointly by the average per-flow throughput and the blocking probability. We present results below in terms of the *fractional average throughput*: the ratio of the average per-flow throughput to the link capacity (100 BUs).

## 5 Simulation Results

We first present results when flows have no rate limitation outside the considered networks. We next study the impact of a limited access rate on the performance of the routing algorithms. In the remainder of the paper, alternative paths for a given flow correspond to the subset of candidate paths with at least one hop more than the minimum-hop path(s).

### 5.1 Comparative Study with no Access Rate Limitation

Figure 4 plots the fractional average throughput as a function of the load, as achieved by WS, MU, SW, WS/TR with a TRL of 0.05 and SW/TR with a TRL of 0.99. Other values of TRL were considered for SW/TR (0.9, 0.5 and 0.1); Figure 6 shows the resulting fractional average throughput for every TRL tested. We also considered two other values of TRL for WS/TR (0.1 and 0.25). We do not show the corresponding throughput results, because hardly any difference was noticed compared to WS/TR(0.05). Table 1 gives the blocking rates produced by all the algorithms.

We first observe that SW/TR(0.99) performs best in terms of blocking and throughput under all load conditions. Note that a TRL of 0.99 in our simulations means that the overflow traffic on a given alternative path is limited to

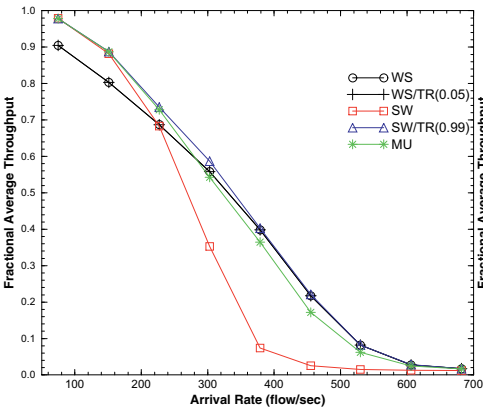


Fig. 4. Throughput comparison with a limited access rate, Full-mesh topology

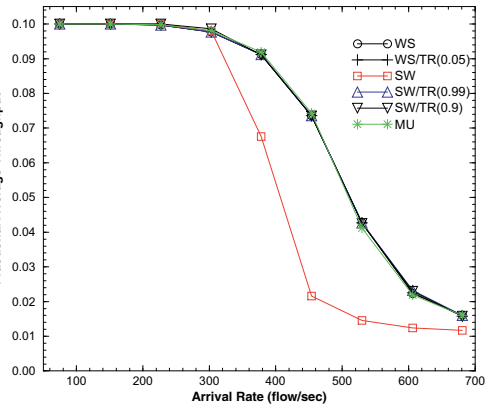


Fig. 5. Throughput comparison with  $r^+ = 10BU s$ , Full-mesh topology

Table 1. Blocking rates, Full-mesh topology

Arrival rate	MU	WS	WS/TR(.05)	WS/TR(.25)	SW/TR(.99)	SW/TR(.5)	SW
530 flows/s	0 %	0%	0%	0%	0 %	0.2%	9.4 %
606 flows/s	5.8%	6.2 %	2.5 %	2.1 %	2.1 %	4.1 %	19.3 %
681 flows/s	17.0%	16.5 %	9.7 %	8.9 %	8.9 %	10.8 %	25.7 %

a single flow. This appears very restrictive and yet yields good performance. The reason is that, SW/TR(0.99) adopts, as the load increases, virtually the same behavior as the *Minimum-Hop Path* algorithm which, on fully-connected uniformly loaded networks, yields the best possible performance at high load. Figure 6 and Table 1 further show that the performance of SW/TR vary significantly with the TRL value. We also noticed that trunk reservation, even with small TRL values, effectively reduces blocking rates compared to that obtained with SW (e.g. 1.78% for a TRL=0.1 at a rate of 530 flows/s). The throughput, however, is only enhanced for high TRL values (e.g. 0.9 and 0.99). MU, on the other hand, performs as well as SW/TR(0.99) at light load, and moves close to WS as the load increases. As far as WS/TR is concerned, figures in Table 1 show that trunk reservation significantly decreases blocking while having very little effect on throughput (same as WS). At low load, trunk reservation is not activated and WS/TR yields the same performance as WS. At overload, both algorithms cannot offer better rates than the minimum rate, however more flows are admitted with trunk reservation.

Results relative to the Europe topology (Figures 7 and 8 for throughput, Table 2 for blocking) highlight a different effect of trunk reservation compared to that obtained on the full-mesh topology. Regarding the performance of SW/TR, we observe that high TRL values (e.g. 0.99 and 0.9) result in a much higher throughput than the other algorithms, but at the cost of higher blocking rates.

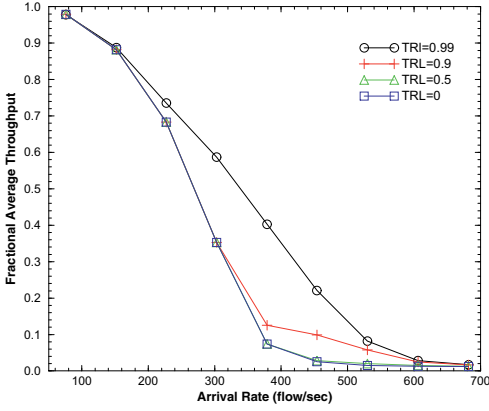


Fig. 6. SW/TR, Full-mesh topology

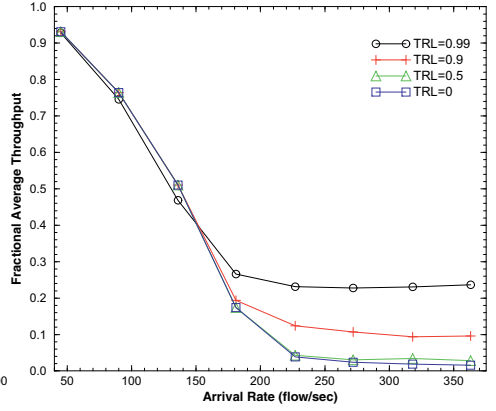


Fig. 7. SW/TR, Europe topology

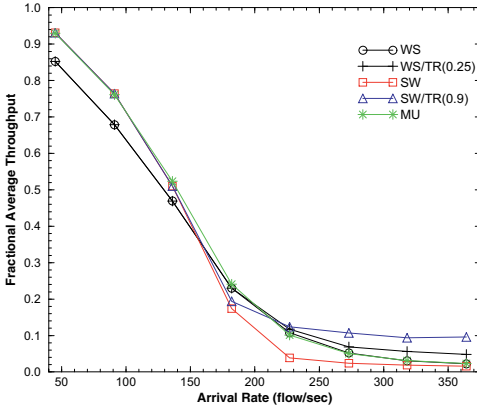


Fig. 8. Throughput comparison, Europe topology

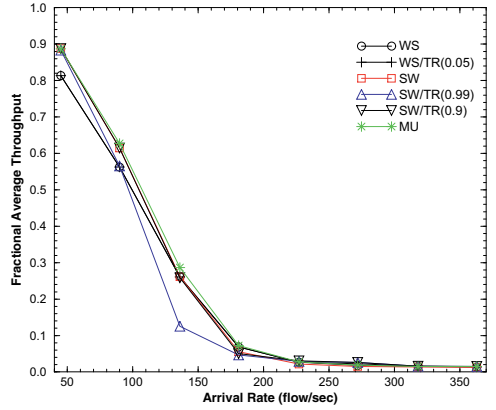


Fig. 9. Throughput comparison, US topology

Table 2. Blocking rates, Europe topology

Arrival rate	MU	WS	WS/TR(.05)	WS/TR(.25)	SW/TR(.99)	SW/TR(.9)	SW
227 flows/s	1.1%	1.2%	1.3%	1.4%	1.8 %	1.3 %	1.0 %
272 flows/s	6.3%	5.9%	6.1%	6.3%	8.4 %	6.5 %	6.3 %
318 flows/s	9.7%	9.9%	10.0%	10.1%	14.6 %	11.4 %	10.0 %
364 flows/s	13.3%	13.0%	13.3 %	13.5%	20.3 %	15.5 %	13.3 %

For WS/TR, slightly higher blocking rates are obtained with trunk reservation than without (WS), together with higher throughput. In particular, WS/TR(0.25) yields at overload, twice as much throughput as WS and slightly higher blocking rates. As expected, for lower TRL values the performance of WS/TR (resp. SW/TR) moves closer to that of WS (resp. SW). We conclude that, on the Europe topology, high values of TRL, by limiting the possibility of alternative routing, limit both the number of simultaneous active flows on alternative paths (higher throughput) and the total number of admitted flows (higher blocking). As with the previous topology, we observe that the performance of SW varies widely as a function of the TRL, while the performance of WS/TR is relatively stable. We also notice that MU adapts well to the Europe topology. MU yields high throughput under light and moderate load conditions, and low blocking under heavy load.

Results relative to the US topology (Figure 9) are fairly consistent with previous observations, though the performance difference between the algorithms is the smallest observed so far. Compared to the previous topologies, the US topology is more loosely meshed. Shortest paths are, therefore, not much more economical in terms of resources than alternative routes and presumably get rapidly saturated as load increases. The use of alternative routes at a relatively early stage explains the convergence in the performance of WS, MU and SW at moderate and high load. Trunk reservation provides WS/TR with a more effective control over network resources at high load. WS/TR(0.05), for instance, yields the lowest blocking rates (14.0% at a rate of 318 flow/sec against 16.0% for WS and 15.5% for MU) and similar throughput. The US topology highlights another effect of trunk reservation when applied to SW. It appears that SW/TR can potentially yield worse results than SW in terms of both blocking and throughput for very high TRL values (e.g. 0.99). Lower TRL values (e.g. 0.9) yield comparable throughput to the other algorithms, but higher blocking persists. This is due to the fact that trunk reservation is triggered earlier on longer routes and causes most flows to be routed on shortest paths only.

## 5.2 Impact of a Limited Access Rate

Previous results (without no access rate limitation) show that WS and WS/TR perform consistently well at high load, but yield poorer throughput than the other algorithms at light load. This difference in performance is likely to disappear if the throughput of elastic flows is limited by their access rate<sup>1</sup>. We evaluate the performance of the five algorithms considering a relatively high access rate  $r^+$  equal to 10 BUs (representing 10% of the link capacity considered in our simulations).

Figure 5 plots the fractional average throughput as a function of the load, achieved by WS, MU, SW, WS/TR with a TRL of 0.05 and SW/TR with TRL values 0.99 and 0.9, on the full-mesh topology. Table 3 gives the corresponding

---

<sup>1</sup> or by another bottleneck link outside the considered network, or indeed by the server providing the transferred document.

**Table 3.** Blocking rates,  $r^+ = 10BU_s$ , Full-mesh topology

Arrival rate	MU	WS	WS/TR(.05)	SW/TR(.99)	SW/TR(.9)	SW
530 flows/s	0%	0%	0%	0 %	0%	9.3%
606 flows/s	6.8%	6.7%	2.7%	2.3 %	2.3 %	19.5%
681 flows/s	17.7%	17.7%	10.2 %	9.5 %	9.4 %	27.2%

blocking rates. We observe that, apart from SW, all other algorithms now yield the same throughput under all load conditions. On the other hand, WS/TR(0.25) continues to give the lowest blocking rates. Throughput and blocking results also show that MU and WS now yield equivalent performance. Furthermore, we observe that a limited access rate has little incidence on the performance of WS/TR and WS at high load. Its impact is all the more important as the algorithm puts more emphasis on load balancing (For instance, SW/TR(0.99) and SW/TR(0.9) now yield the same throughput).

Results on the Europe and US topologies (not presented here) confirm that WS and WS/TR perform as well as MU at light load, while WS/TR maintains its performance advantage over WS under overload.

## 6 Conclusion

The main intent of this paper is to evaluate the impact of trunk reservation when used for elastic flow routing. From the algorithms *Widest-Shortest Path* and *Shortest-Widest Path*, whose performance advantages and disadvantages are well known, we derived two novel algorithms that integrate a form of trunk reservation. The latter algorithms were compared to the *Maximum Utility Path* algorithm which uses a load dependent utility function for path selection.

Simulation results show that the performance of SW/TR strongly depends on the TRL value and the topological characteristics. Although SW/TR is potentially capable of efficiently handling traffic at low and high load, the choice of the appropriate trunk reservation level remains a serious concern.

WS/TR, in contrast to SW/TR, exhibits relatively stable performance. The TRL values tested in the range of 0.05 to 0.25 yield comparable performance. Although the effect of trunk reservation on elastic flow QoS may depend on topological characteristics, WS/TR generally yields higher overall performance than WS at overload. It seems that the choice of the TRL for WS/TR is less critical than in circuit-switched networks where a TRL which is set too high causes higher blocking. In networks carrying elastic traffic, if flows are unnecessarily blocked, this still benefits ongoing flows. Simulation results show that the poor performance of WS/TR is no longer an issue when elastic flows have a limited access rate, making WS/TR rather attractive. Furthermore, algorithms like WS/TR that select minimum-hop routes in the first place, and that activate trunk reservation at overload generally yield good performance for stream-like



flows. Thus, in a multiservice context, the same WS/TR algorithm could be used for routing both elastic flows and stream flows.

MU, on the other hand, generally adapts well to any network configuration, but its behavior at overload seems perfectible. Under heavy traffic conditions, MU generally yields higher blocking than WS/TR. MU does not always provide the same level of protection against performance degradation in overload as trunk reservation.

Trunk reservation with a fixed trunk reservation level is not an adaptive mechanism. In our future studies we will continue the exploration of adaptive routing schemes whose parameter(s) can be automatically adjusted to account for different network topologies and changing traffic conditions.

## References

1. Sara Oueslati-Boulahia and Eliane Oubagha. An Approach to Routing Elastic Flows. In *ITC 16*. Elsevier, June 1999.
2. Z. Wang and J. Crowcroft. Quality-of-Service Routing for Supporting Multimedia Applications. *IEEE JSAC*, 14(7):1288–1294, September 1996.
3. George Apostolopoulos, Roch Guerin, Sanjay Kamat, and Satish K. Tripathi. Quality of Service Based Routing : A Performance Perspective. In *Sigcomm '98*, August 1998.
4. R. Gawlick, A. Kamath, S. Plotkin, and K.G. Ramakrishnan. Routing and Admission Control in General Topology Networks. Technical report, Stanford University, 1995.
5. L. Massoulié and J.W. Roberts. Arguments in Favor of Admission Control for TCP flows. In *ITC16*. Elsevier, June 1999.
6. L. Massoulié and J.W. Roberts. Bandwidth Sharing and Admission Control for Elastic Traffic. *ITC Seminar, Yokohama*, October 1998. to appear in *Telecommunication Systems*.
7. Qingming Ma. *Quality-of-Service Routing in Integrated Service Networks*. PhD thesis, Carnegie Mellon University, Pittsburgh, January 1998.
8. M. Mellia C. Casetti, G. Favalessa and M. Munafo. An Adaptive Routing Algorithm for Best-Effort Traffic in Integrated-Services Networks. In *ITC 16*. Elsevier, June 1999.
9. Qingming Ma and Peter Steenkiste. Routing Traffic with Quality-of-Service Guarantees in Integrated Services Networks. In *NOSSDAV'98*, July 1998.
10. James W. Roberts. Realizing Quality of Service Guarantees in Multiservice Networks. In T. Horsegawa et al., editor, *PMCCN'97*. Chapman and Hall, 1998.
11. Sara Oueslati-Boulahia and Eliane Oubagha. A Comparative Studing of Routing Algorithms for Elastic Flows in a Multiservice Network. Technical Report DE/DAC/OAT/79, CNET-France Telecom, September 1999.
12. J.M. Akinpelu. The overload performance of engineered networks with non-hierarchical and hierarchical routing. In *ITC-10, Montreal, Canada*, 1983.
13. Ren P. Liu and Peter J. Moylan. Dynamic Trunk Reservation for Teletraffic Links. In *GLOBCOM'95*, volume 1, pages 432–436, 1995.
14. D. Mitra and R.J. Gibbens. State-dependent routing on symmetrical loss networks with trunk reservation. *IEEE Trans. Commun.*, 41(2):400–411, Febuary 1993.

# New Distributed Multicast Routing and Its Performance Evaluation

Takuya Asaka<sup>123</sup>, Takumi Miyoshi<sup>23</sup>, and Yoshiaki Tanaka<sup>23</sup>

<sup>1</sup> NTT Service Integration Laboratories

9-11, Midori-Cho 3-Chome, Musashino-Shi, Tokyo, 180-8585 Japan.

<sup>2</sup> Global Information and Telecommunication Institute, Waseda University

3-10, Nishi-Waseda 1-Chome, Shinjuku-Ku, Tokyo, 169-0051 Japan.

<sup>3</sup> Okinawa Research Center

Telecommunications Advancement Organization of Japan

21-1, Nishi-Waseda 1-Chome, Shinjuku-Ku, Tokyo, 169-0051 Japan.

**Abstract.** With conventional dynamic routing algorithms, many query messages are required in a distributed environment for efficient multicast routing of any traffic volume. We have developed a dynamic routing algorithm that uses a predetermined path search in which an appropriate multicast path is dynamically constructed by searching only a few nodes. This algorithm can construct an efficient multicast tree for any traffic volume. Simulation has shown that the proposed algorithm is advantageous compared with conventional dynamic routing algorithms when nodes are added to or removed from the multicast group during steady-state simulation.

## 1 Introduction

Future computer-network applications such as teleconferencing or remote collaboration will rely on the ability of networks to provide multicast services. Multicasting is expected to become widely used [1,2,3], and is well suited to these services because it uses network resources efficiently. In multicasting, a point-to-multipoint (multicast) connection is used to copy packets only at branching nodes, which ensures network efficiency. Naturally, the smallest possible amount of network resources should be used to set up the multicast connection.

Multicast routing problems are either *static* or *dynamic*. In *static* routing problems, the members of the multicast group remain unchanged during the lifetime of the multicast connection. In *dynamic* routing problems, members can join or leave the group during the lifetime of the connection. Dynamic multicast routing is important for actual multicast applications and is supported by protocols in ATM network [3] and Internet protocols [4,5,6].

Here, we focus on a dynamic multicast routing algorithm that can satisfy the following requirements.

- (1) *Minimized average multicast tree cost.* Every link has a cost (or a metric), and the tree cost is the sum of the costs for all links included in the multicast

tree. Minimizing the tree cost ensures efficient use of bandwidth. This is called the dynamic Steiner problem.

- (2) *Scalability in a distributed environment.* A central server should not be used to control joining nodes because a large-scale network could overload the server, thus degrading performance. Moreover, minimizing the overhead of nodes is important from the viewpoint of algorithm scalability even if there is no central server does.
- (3) *Robustness against the number of joining nodes.* The number of joining nodes strongly affects the performance of conventional algorithms. The performance should be independent of the number of joining nodes.
- (4) *Minimized worst-case cost of the multicast tree.* The worst-case cost produced by the algorithm should be theoretically bounded as small as possible.

Many dynamic multicast routing algorithms have been proposed [4,5,6,7,8,9] [10,11,12,13,14]. However, none of these algorithms can satisfy all of the above requirements. The greedy algorithm [8,9] selects the shortest path to an existing multicast tree when a node is added. It can construct a near-optimal multicast tree, but requires many query/reply messages between nodes when implemented in a distributed environment [13,14]. Thus, the algorithm is not scalable in a distributed environment. The pruned shortest-path tree algorithm [4,5,6,7] finds the shortest path from the source node (or the center node) to the nodes in the multicast group when a node is added to the multicast tree. This algorithm cannot construct an appropriate multicast tree, though, from the viewpoint of tree cost. The virtual trunk dynamic multicast (VTDM) routing algorithm [10] constructs multicast trees based on the virtual trunk, which is the tree of the underlying graph. However, the “trunk number” for constructing the virtual trunk must be determined according to the estimated number of nodes that will join and it is not flexible.

We have developed a dynamic routing algorithm that can satisfy all four of the above requirements. It uses a predetermined path search where only a few nodes are searched to determine to which existing node a newly added node should be connected. In this algorithm, the searched nodes are on predetermined paths: the paths from new added nodes to the source node on the minimum-spanning tree and the shortest-path tree. The node and network overheads are small because only a few nodes are searched. The performance is similar to the greedy algorithm but does not depend on the number of joining nodes. Our simulation has shown that our algorithm is advantageous when nodes are added to or removed from the multicast group in the steady state. We discuss the competitive ratio of our algorithm, and show its advantage over the greedy algorithm.

## 2 Problem Definition

To define the dynamic multicast routing problem formally from the viewpoint of minimizing multicast tree cost, we use the terminology of graph theory for

the models. In the model used for the conventional dynamic multicast routing problem, the network is modeled as a graph whose edges have costs. If nodes can be added or removed during the lifetime of the multicast connection, this problem becomes the dynamic multicast routing problem, i.e., the dynamic Steiner tree problem. Let  $R = \{r_1, r_2, \dots, r_K\}$  be a sequence of requests, where  $r_i$  is either adding or removing a destination node to or from the multicast group. Let  $S_i$  be the set of nodes in the multicast group after request  $r_i$  has been made. In response to request  $r_i$ , multicast tree  $T_i$  is constructed using a dynamic multicast routing algorithm. The dynamic multicast routing problem can thus be formally defined as follows.

Given graph  $G = (V, E)$ , a nonnegative weight for each  $e \in E$ , and  $Z \subseteq V$ , and a sequence  $R$  of requests, find a sequence of multicast trees  $\{T_1, T_2, \dots, T_K\}$  in which  $T_i$  spans  $Z_i$  and has minimum cost.

The dynamic multicast routing problem considered in this paper does not allow re-routing of existing connections when additional requests are received. One node is the source for a multicast communication, and this node cannot be removed from the multicast group during the life of the multicast connection.

In this paper, vertices that are included in a multicast tree are called existing nodes, and the source node is an existing node. Vertices that do not participate in a multicast group but that are included in the multicast tree are called intermediate nodes. Moreover, vertices that are neither an existing nor an intermediate node are called non-existing nodes.

### 3 Conventional Algorithms

Several dynamic multicast routing algorithms have been reported [8,9,10,11,12]. As mentioned earlier, the greedy algorithm [8,9] selects the shortest path to an existing multicast tree when adding a node. The shortest path between a pair of nodes can be calculated using Dijkstra's algorithm before routing. In the greedy algorithm, selecting a new path is the best way to add a node. However, when greedy algorithm is implemented in a distributed environment, a newly added node must flood the entire network with query messages [13]. That is, a newly added node sends a query message to all its neighbors on the shortest-path tree rooted at the node, and this continues until the query message reaches either an existing node or a leaf node of the shortest-path tree. Consequently, there is a large processing overhead for the query message in each node. The network overhead becomes particularly large when there are many non-existing nodes. As another approach [15], a manager router is introduced to control the addition and removal of nodes. However, the processing overhead of the manager router is high when there are many nodes in the network, and a failure of the manager router would cause a fatal error in the construction of the multicast tree.

As mentioned above, the pruned shortest-path tree algorithm (pruned SPT) [4,5,6,7] finds the shortest path from a source node [7] (a "core" in CBT [4]

or a “rendezvous point” in PIM-SM [5]) to the nodes in the multicast group when a node is added to the multicast tree. The pruned SPT algorithm has been implemented as an actual routing protocol [4,5,6] because it is easier to implement than the greedy approach. The shortest t-path tree is spanned from a source node (or a core or rendezvous point) to every node in the multicast group. Moreover, the multicast tree is obtained by deleting nonessential edges. The cost of a multicast tree made using this approach is higher [16] than for one made using the greedy approach.

The virtual trunk dynamic multicast (VTDM) routing algorithm [10] constructs multicast trees based on the virtual trunk, which is the tree of the underlying graph. The virtual trunk and the multicast tree that uses it are constructed as follows:

- Step 1:** Find the shortest paths for all pairs of nodes. Count the number of shortest paths passing through each node.
- Step 2:** Define a set  $F$  as vertices having the top  $\lceil \theta N \rceil$  largest numbers; these vertices are called virtual-trunk nodes.
- Step 3:** Construct a complete graph for  $F$  by replacing the shortest paths between pairs of vertices in  $G$  with distances between pairs of vertices. Find the minimum-spanning tree for the complete graph.
- Step 4:** Convert edges in the minimum-spanning tree back to the corresponding shortest paths in the graph  $G$ . Run the minimum-spanning tree algorithm and remove any unnecessary nodes or links. The obtained tree is called the virtual trunk.
- Step 5:** Connect nodes not in the virtual trunk to the nearest node in the virtual trunk.

$N$  is the number of nodes in graph  $G$ , and parameter  $\theta$  ( $0 \leq \theta \leq 1$ ) determines the number of trunk nodes. When  $\theta = 0$ , the VTDM algorithm is the same as the pruned SPT because only the source node is selected as a trunk node. Similarly, when  $\theta = 1$ , the VTDM algorithm is the same as the pruned minimum-spanning tree (pruned MST), where the MST is used instead of the SPT as in the pruned SPT. In this algorithm,  $\theta$  must be determined according to the estimated number of nodes that will be join. However, actual traffic estimation is difficult, and setting  $\theta$  is a complicated task for network management.

The other conventional dynamic multicast routing algorithms take different approaches. For example, some allow re-routing of multicast connections [11,12,17].

## 4 Dynamic Multicast Routing Algorithm Using Predetermined Path Search

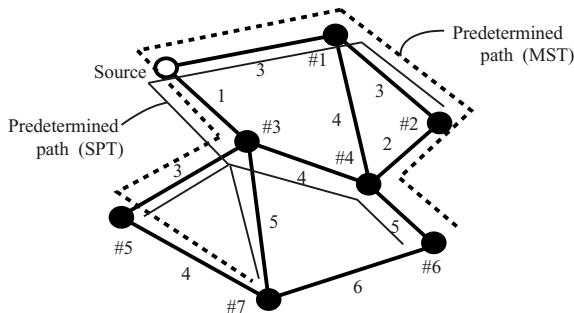
Our algorithm is based on the greedy algorithm and is intended to have the same performance. It uses *query* and *reply* messages to obtain membership information, as does the greedy algorithm. Thus, this approach does not require

a manager server for multicast routing. Furthermore, our dynamic multicast algorithm searches only a few nodes to determine which existing node should be connected. A newly added node connects to this existing node using the shortest path. The node and network overheads are small because the number of searched nodes is restricted. An appropriate path is selected based on the current multicast tree. Thus, our algorithm is suitable for a distributed environment.

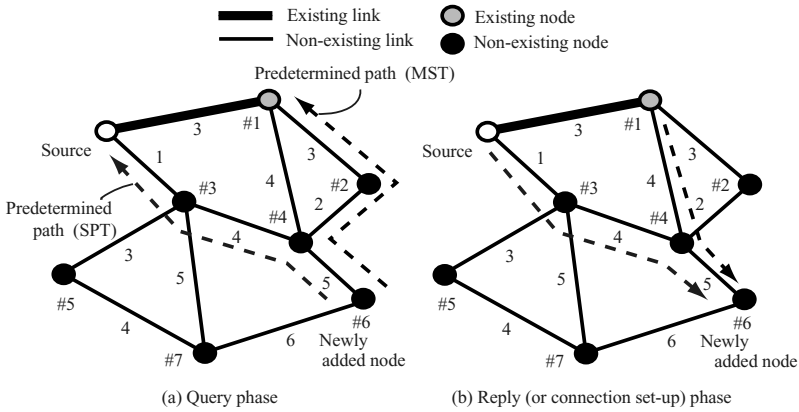
In this algorithm, two kinds of spanning trees are prepared for the predetermined paths: a minimum-spanning tree and a shortest-path tree having a source node as a root. Figure 1 shows examples of the MST and SPT. The minimum-spanning tree can be calculated using either Kruskal's or Prim's algorithm with computational complexity  $O(m \log n)$ , and the shortest-path tree can be calculated using Dijkstra's algorithm with computational complexity  $O(m + n \log n)$ . Our algorithm works as follows.

- Step 1:** A newly added node sends query messages to nodes on two predetermined paths toward the source.
- Step 2:** If a queried node is an existing node, it sends a reply message to the new added node and the query message is not transmitted to the next node. Otherwise, the query message is transmitted to the next node on the predetermined path.
- Step 3:** After the added node receives reply messages from existing nodes on each predetermined path, it connects to the closest node among the nodes that sent reply messages.

An example of how this procedure works is shown in Fig. 2. The newly added node #6 sends query messages to nodes on two predetermined paths toward the source. First, we describe the case of using the MST as a predetermined path. In Fig. 2, only node #1 is an existing node. The predetermined path for a newly added node is  $\langle \#6 \rightarrow \#4 \rightarrow \#2 \rightarrow \#1 \rightarrow \text{source} \rangle$ . When a new node requests to join, a query message is transmitted to node #6. Since node #6 is not an existing node, the query message is retransmitted to the next node. This retransmission continues until node #1 receives the query message. Node #1 is an existing node, so it sends a reply message to the newly added node. Next, we describe



**Fig. 1.** Example of MST and SPT.



**Fig. 2.** Dynamic multicast routing algorithm using predetermined path search.

the case of using the SPT as a predetermined path. The predetermined path for a newly added node is  $\langle \#6 \rightarrow \#4 \rightarrow \#3 \rightarrow \text{source} \rangle$ . The algorithm works similar to the case of MST and the newly added node receives a reply message from the source. Thus, the newly added node #6 receives two reply messages from node #1 and from the source node. Node #1 is closer to the newly added node than the source, so the newly added node is connected to node #1 using the shortest path between them.

In this algorithm, the added node should be connected to the source node using the path that is nearly the shortest path when there are few existing nodes. On the other hand, when there are many existing nodes, the added node should be connected to the existing node using nearly the minimum spanning.

Moreover, the node and network overheads are lower than with the greedy algorithm. This is because only nodes on predetermined paths are searched and only these nodes receive query messages. In the example in Fig. 2, the proposed algorithm needs six query messages. These messages are  $\langle \#6 \rightarrow \#4 \rangle$ ,  $\langle \#4 \rightarrow \#2 \rangle$ ,  $\langle \#2 \rightarrow \#1 \rangle$ ,  $\langle \#6 \rightarrow \#4 \rangle$ ,  $\langle \#4 \rightarrow \#3 \rangle$  and  $\langle \#3 \rightarrow \text{source} \rangle$ . The greedy algorithm, on the other hands, needs seven. These are  $\langle \#6 \rightarrow \#4 \rangle$ ,  $\langle \#4 \rightarrow \#2 \rangle$ ,  $\langle \#2 \rightarrow \#1 \rangle$ ,  $\langle \#4 \rightarrow \#3 \rangle$ ,  $\langle \#3 \rightarrow \text{source} \rangle$ ,  $\langle \#6 \rightarrow \#7 \rangle$ ,  $\langle \#7 \rightarrow \#5 \rangle$ . Similarly, if node #4 instead of node #6 is newly added to the multicast group, the proposed algorithm needs four and the greedy algorithm needs seven. As the network size increases, the advantage of the proposed algorithm increases because the greedy algorithm broadcasts query messages.

Another version of this algorithm uses only the MST as the predetermined path, rather than both the MST and SPT. This does not require reply messages from the existing nodes because a connection set-up message can be used instead. Since the newly added node need not determine which existing node is nearer. However, the tree cost may be higher than in the previous case. There is a trade-off relationship between processing overhead and cost performance.

**Table 1.** Parameters used for simulations.

Parameter Value		Parameter Value	
$N$	50	$\bar{e}$	3
$\alpha$	0.25	$k$	25
$\beta$	0.20	$\mu$	0.9
$\gamma$	0.20		

## 5 Simulation Model

We evaluated the performance of our algorithm through simulation, using the same model and parameters that were used by Lin and Lai [10]. A network is modeled as a random graph possessing some of the characteristics of an actual network [7]. The vertices representing nodes are randomly distributed on a rectangular coordinate grid, and each vertex has integer coordinates. For a pair of vertices, say  $u$  ( $0 \leq u \leq 1$ ) and  $v$  ( $0 \leq v \leq 1$ ), an edge is added according to the following probability:

$$P_e(u, v) = \frac{k\bar{e}}{N} \beta \exp \frac{-d(u, v)}{L\alpha}, \quad (1)$$

where  $N$  is the number of vertices in the graph,  $\bar{e}$  is the mean number of degrees of a vertex,  $k$  is a scale factor related to the mean distance between two vertices,  $d(u, v)$  is the Euclidean distance between vertices  $u$  and  $v$ ,  $L$  is the maximum distance between any two vertices in the graph, and  $\alpha$  and  $\beta$  are parameters (real numbers between 0 and 1). The edge density is increased by increasing the value of  $\beta$ . The edge density of shorter edges relative to that of longer ones is decreased by decreasing the value of  $\alpha$ . Setting the values of  $\alpha$  and  $\beta$  to 0.25 and 0.20, respectively, generates a graph that roughly resembles a geographical map of the major nodes in the Internet [7]. Once the vertices and edges have been generated, we can be sure that the graph is composed of only one component.

In the simulations, requests to add or remove a node to/from the multicast group are periodically generated. We used a probability model to generate the sequence of requests, i.e., to determine whether each request was to add or remove. The probability for adding a node was determined by

$$P_c(q) = \frac{\gamma(N - q)}{\gamma(N - q) + (1 - \gamma)q}, \quad (2)$$

where  $q$  is the current number of nodes in the multicast group and  $\gamma$  ( $0 \leq \gamma \leq 1$ ) is a parameter (a real number). That determines the size of the multicast group in equilibrium. Each node had a different rate for joining the multicast, i.e., a different probability that the  $i$ th node would be selected to join. We defined  $A$  as a set of non-joining nodes and the joining rate of the  $i$ th node was given by

$$P_{join}(i) = \begin{cases} (1 - \mu)\mu^{i-1}/F_0, & \text{for } i \in A \\ 0, & \text{for } i \notin A \end{cases} \quad (3)$$



where  $F_0 = \sum_{i \notin A} (1 - \mu) \mu^{i-1}$  and a parameter  $\mu$  ( $0 \leq \mu \leq 1$ ) determined the bias of the joining rate. In simulations, if a joining event is generated with  $P_c(q)$ , a node that will join is determined with  $P_{join}(i)$ . If a removal event is generated with  $P_c(q)$ , a node that will be removed is randomly determined.

We compared our algorithm with four conventional algorithms: the greedy algorithm, the pruned SPT, the VTDM algorithm, and the pruned MST. For the VTDM algorithm, parameter  $\theta$  was set to 0, 0.2, 0.4, or 1. The VTDM algorithm with  $\theta = 0$  corresponded to the pruned SPT, and with  $\theta = 1$  it corresponded to the pruned MST.

In the simulations, the average multicast tree cost was used as a measure of performance. We generated ten different networks and calculated the average tree cost. Each multicast connection consisted of a sequence of 20,000 requests to add or remove nodes. The costs for the first 2,000 requests were not used in calculating the average costs to eliminate the effect of the initial conditions in the simulation. Table 1 shows the default values of the simulation parameters.

## 6 Simulation Results

### Basic cases

The relationship between the tree cost and the average multicast group size is shown in Fig. 3 for 20 and 50 nodes, where the multicast tree costs are normalized by the costs with the greedy algorithm. In both cases, the cost with the proposed algorithm was close to that with the greedy algorithm. The reason is as follows. In the proposed algorithm, the selected node on the tree and the newly added node are connected using the shortest path. Thus, the proposed algorithm works like the greedy algorithm. Moreover, the proposed algorithm can find an appropriate existing node on predetermined paths. When the multicast group is large, an existing node on the MST is likely to be selected as the node to be connected for the newly added node. On the other hand, when the multicast group is small, an existing node on the SPT is likely to be selected as the node to be connected for the newly added node. Thus, the number of wasteful searches is small.

Furthermore, the proposed algorithm is slightly superior to the greedy algorithm when the group size is small. In these cases, the greedy algorithm may cause the construction of roundabout routes since many nodes are removed. The SPT as the predetermined path can prevent the construction of these routes because nodes nearby the source are selected as nodes to be connected.

The other algorithms, however, could not match the performance achieved with the greedy algorithm for any multicast group size. These algorithms were especially poor for extremely small group sizes, and require that the algorithm or parameters be selected according to the group size. In comparison, our algorithm worked well for any multicast group size without requiring a parameter for control, and was not greatly influenced by the number of nodes.

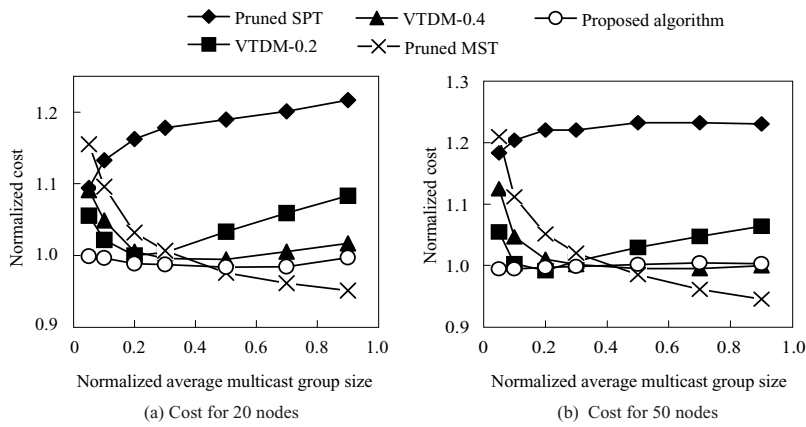


Fig. 3. Cost of average multicast tree with the conventional and proposed algorithms.

Traffic bias

The relationship between the tree cost and the average multicast group size is shown in Fig. 4 when the joining rate of each node was varied. The traffic bias was small when parameter  $\mu$  for the joining rate was large, and each node had the same rate when  $\mu = 1$ .

The tree cost of all the algorithms became worse than that with the greedy algorithm, but the proposed algorithm was slightly more robust for traffic bias than the other algorithms. This is because the proposed algorithm - just like the greedy algorithm - can construct multicast trees depending on existing nodes.

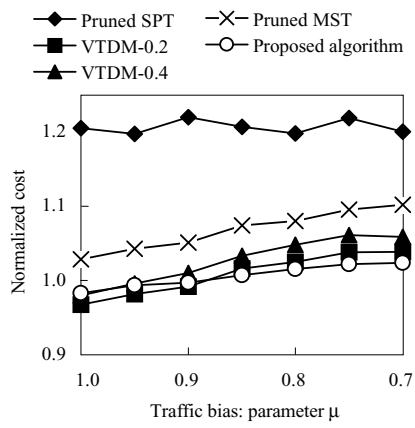


Fig. 4. Comparison of average multicast tree cost at different traffic biases.

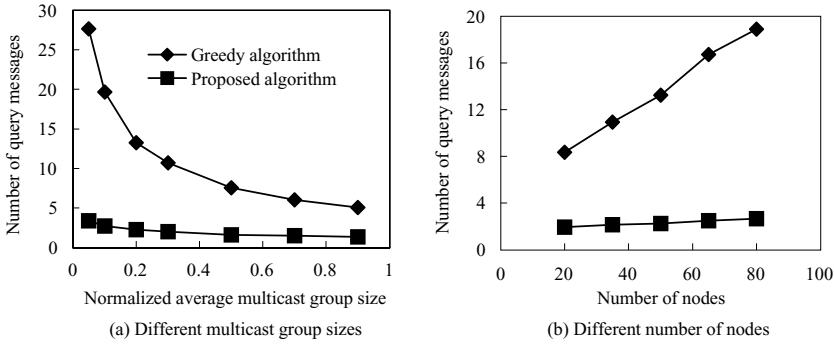


Fig. 5. Number of query messages with the proposed and greedy algorithms.

Number of query messages

The number of query messages with the proposed and greedy algorithms [13] is compared in Fig. 5. In this figure, the number of query messages is the total number of query messages received by all nodes when a node is newly added to the multicast group.

When the multicast group size is small, more messages were needed with the greedy algorithm (Fig. 5(a)). This is because the greedy algorithm requires query message transmission until the query message reaches either an existing node or a leaf node of the shortest-path tree. When the number of nodes was increased, the number of query messages increased with both algorithms (Fig. 5(b)). However, the increase was smaller with the proposed algorithm than with the greedy algorithm. Thus, the loads offered to nodes and the network should always be smaller with the proposed algorithm than with the greedy algorithm.

7 Competitiveness Analysis

Here, we discuss the *competitive ratio* of the proposed algorithm, defined as the maximum ratio of the cost of the algorithm over the optimal one. The competitive ratios of conventional algorithms and the proposed algorithm are shown in Table 2. In the static case, the members of the multicast group remain unchanged during the lifetime of the multicast connection. In the join-only case, the members of the multicast group do not quit a session. In the join-remove case, the members of the multicast group join and quit a session during the lifetime of the multicast connection. We denote the number of participant nodes as  $M$  in the static case, and the maximum number of participant nodes in the lifetime of a multicast connection in the join-only and join-remove cases.

The competitive ratios of conventional algorithms are described in [18,19]. The competitive ratio of the greedy algorithm had not been derived in the join-remove case, but the lower bound of the competitive ratio had been derived. For the proposed algorithm, the competitive ratio is  $\max(N - M, M)$  since the worst case is always bounded by either the pruned MST or the pruned SPT.

**Table 2.** Competitive ratios.

	Static	Join-only	Join-remove
Pruned MST	$N - M$	$N - M$	$N - M$
Pruned SPT	$M$	$M$	$M$
Greedy algorithm	$2 - 2/M$	$\log M$	$2^M$ (*)
Proposed algorithm	$\max(N - M, M)$	$\max(N - M, M)$	$\max(N - M, M)$
			(*)lower bound

Table 2 shows that the greedy algorithm has an advantage over the other algorithms in the static and join-only cases. On the other hand, the greedy algorithm fails dramatically in the join-remove problem, while the other algorithms do not become any worse than in the static and join-only cases. Although our algorithm does not have an advantage over the pruned MST and the pruned SPT in all cases, it has a large advantage over the greedy algorithm.

## 8 Conclusion

We proposed a dynamic routing algorithm that uses a predetermined path search, in which an appropriate multicast path is dynamically constructed by searching only a few nodes. Simulation showed that the performance of this is close to that of the greedy algorithm and that it is superior when nodes are added to or removed from the multicast group during steady-state simulation. The node overhead is lower than with the greedy algorithm since it requires only a few query messages, so our algorithm is suitable for a distributed environment. Moreover, it can construct an efficient multicast tree that is independent of the multicast group size. We also showed that the competitive ratio of our algorithm is superior to that of the greedy algorithm.

Some research problems remain concerning multicasting using this method. One is the performance of multicasting under diverse practical traffic patterns and network topologies. The development of a multicast routing protocol scheme also deserves attention.

## References

1. The IP Multicast Initiative, The IP Multicast Initiative Home page, <http://www.ipmulticast.com/>, Dec. 1998.
2. The MBone Information Web, Home Page, <http://www.mbone.com/>, Dec. 1998.
3. C. Diot, W. Dabbous and J. Crowcroft, "Multipoint Communication: a Survey of Protocols, Functions, and Mechanisms," IEEE JSAC., Vol. 15, No. 3, pp. 277-290, April 1997.
4. A. Ballardie, "Core Based Trees (CBT Version 2) Multicast Routing - Protocol Specification -," RFC2189, Sept. 1997.

5. D. Estrin, D. Farinacci, A. Helmy, D. Thaler, S. Deering, M. Handley, V. Jacobson, C. Liu, P. Sharma and L. Wei, "Protocol Independent Multicast-sparse Mode (PIM-SM): Protocol Specification," RFC2362, June 1998.
6. J. Moy, "Multicast Extensions to OSPF," RFC 1584, March 1994.
7. M. Doar and I. Leslie, "How Bad is Naïve Multicast Routing?" IEEE INFOCOM'93, pp. 82–89, March 1993.
8. B. M. Waxman, "Routing of Multipoint Connections," IEEE JSAC., Vol. 6, No. 9, pp. 1617–1622, Dec. 1988.
9. B. M. Waxman, "Performance Evaluation of Multipoint Routing Algorithms," IEEE INFOCOM'93, pp. 980–986, March 1993.
10. H. Lin and S. Lai, "VTDM - a Dynamic Multicast Routing Algorithm," IEEE INFOCOM'98, pp. 1426–1432, March–April 1998.
11. J. Kadirire, "Minimizing Packet Copies in Multicast Routing by Exploiting Geographic Spread," ACM SIGCOMM Computer Communication Review, Vol. 24, pp. 47–63, 1994.
12. J. Kadirire, "Comparison of Dynamic Multicast Routing Algorithms for Wide-area Packet Switched (Asynchronous Transfer Mode) Networks," IEEE INFOCOM'95, pp. 212–219, March 1995.
13. R. Venkateswaran, C. S. Raghavendra, X. Chen and V. P. Kumar, "DMRP: a Distributed Multicast Routing Protocol for ATM Networks," ATM Workshop'97, May 1997.
14. K. Carlberg and J. Crowcroft, "Building Shared Trees using a One-to-many Joining Mechanism," ACM Computer Communication Review, Vol. 27, No. 1, pp. 5–11, Jan. 1997.
15. M. Faloutsos, A. Banerjea, and R. Pankaj, "QoSMIC: Quality of Service Sensitive Multicast Internet Protocol," ACM SIGCOMM '98, Sep. 1998.
16. L. Wei and D. Estrin, "The Trade-offs of Multicast Trees and Algorithms," ICCN'94, Sept. 1994.
17. F. Bauer and A. Varma, "ARIES: a Rearrangeable Inexpensive Edge-based On-line Steiner Algorithm," IEEE INFOCOM'96, pp. 361–368, March 1996.
18. M. Faloutsos, R. Pankaj and K. C. Sevcik, "Bounds for the On-line Multicast Problem in Directed Graphs," 4th International Colloquium on Structural Information and Communication Complexity, July 1997.
19. P. Winter, "Steiner Problem in Networks: a Survey," Networks, Vol. 17, pp. 129–167, 1987.

# Distance-Vector QoS-Based Routing with Three Metrics <sup>★</sup>

Luís Henrique M. K. Costa<sup>1,2</sup>, Serge Fdida<sup>1</sup>, and Otto Carlos M. B. Duarte<sup>2</sup>

<sup>1</sup> LIP6 – Université Pierre et Marie Curie  
4, place Jussieu - 75252 - Paris Cedex 05 - France  
{Luis.Costa,Serge.Fdida}@lip6.fr

<sup>2</sup> Grupo de Teleinformática e Automação – GTA  
Universidade Federal do Rio de Janeiro – COPPE/EE  
P.O. Box 68504 - 21945-970 - Rio de Janeiro - RJ - Brasil  
otto@gta.ufrj.br

**Abstract.** Quality of Service (QoS) based routing provides QoS guarantees to multimedia applications and an efficient utilization of the network resources. Nevertheless, QoS routing is likely to be a costly process that does not scale when the number of nodes increases. Thus, the routing algorithm must be simple and a class-of-service routing is an alternative to provide QoS guarantees instead of per-flow routing. This paper proposes and analyzes the performance of a distance-vector QoS routing algorithm that takes into account three metrics: propagation delay, available bandwidth, and loss probability. Classes of service and metric-combination are used to turn the algorithm scalable and as complex as a two-metric one. The processing requirements of the proposed algorithm and those of an optimal approach are compared and the results show an improvement up to 50%.

## 1 Introduction

To facilitate the use of multimedia applications in the Internet, new characteristics and services must be added. New classes of service should be offered to provide better guarantees of Quality of Service (QoS). An important issue for that purpose is the definition of a routing architecture that considers the QoS requirements of applications. In traditional routing, packets are delivered using a route based on their source and destination addresses while in QoS-based routing the traffic requirements are also taken into account by the routing algorithm.

In classical routing, the number of hops is the metric used to make route decisions. QoS-based routing is more complex because the routing decisions are based on  $n$  metrics. Scalability is a main issue of QoS-based routing [1,2]. The routing algorithm must be simple. Some authors consider the use of precomputed paths to minimize the setup overhead [2,3]. Scalability also relates to the size of routing tables, important even in traditional routing. The uncertainty of QoS

---

<sup>★</sup> Sponsors: UFRJ, CNPq, FAPERJ, CAPES, REENGE, and COFECUB.

parameters may also affect QoS routing, e.g., in a link-state protocol the network topology database may be out of date [4,5]. The routing algorithm may take into account these uncertainties. Setting the period of update messages is a tradeoff between the accuracy of QoS parameters and the protocol overhead [2,6].

QoS routing is likely to be an expensive process [2] – it should not keep per-flow state in routers, as this approach is unscalable [7,8]. A more suitable solution is to use per-flow state only for special flows (flows that have stringent QoS requirements which demand a value-added service). Less demanding flows can be satisfied with a connectionless service, where routing is done per class of service and not per flow. In this work a class of service is defined by specific limits on a set of QoS requirements, e.g., end-to-end delay and available bandwidth. The application chooses the class of service that fits the best its requirements.

This paper discusses the approaches found in the literature and proposes an algorithm for unicast route computation based on distance-vectors. The algorithm is based on the SMM approach first presented in [9], where the source routing version was proposed. This paper presents the distance-vector algorithm (SMM-DV) and results obtained from simulation. SMM-DV was implemented as an additional module to NS-2 (Network Simulator) [10]. The performance analysis compares SMM-DV with an optimal algorithm for three-metric route computation. The objective is to evaluate the processing requirements of the proposed algorithm. The results show an improvement up to 50% for SMM-DV.

The rest of the paper is organized as follows. Sections 2 and 3 discuss the main issues related to QoS routing. Section 4 presents our SMM-DV proposal. The proposed approach was implemented and tested, as shown in Section 5. Section 6 presents some final considerations.

## 2 Routing Model

Several routing protocols were developed for packet switched networks [11]. They are classified in distance-vector based (DV) and link-state based (LS), and they use variants of the Bellman-Ford and of the Dijkstra algorithm. There are two approaches for packet forwarding: source routing and hop-by-hop routing. In the first one, the packet header has a list of routers that must be traversed from the source to the destination (the source chooses the route). In hop-by-hop routing, each router chooses the next hop that a packet will follow. The routing model impacts QoS-based routing, as discussed in the next two sections.

### 2.1 Source Routing Versus Hop-by-Hop Routing

Since QoS requirements are diverse, it is difficult to specify a set of requirements that applies to most applications. Therefore source routing, which computes forwarding paths on-demand on a per-flow basis, fits well. QoS-based source routing can be thought as a way to implement QoS paths for specific applications in enterprise networks. The source-routing concept is present in explicit routing [12], which provides fixed paths to applications.

On the other hand, hop-by-hop routing allows distributed computation, produces smaller overhead, and reduces setup delays. It is currently the most common forwarding paradigm whereas source routing is only used for specific applications. This is due also to security issues, since to setup a route at the source one must know the entire topology. In general this information is not available, as many organizations may not want to divulge their internal topology.

## 2.2 Dv versus LS Routing

In DV routing each node knows its direct neighbors and keeps a database with the distance (hop count) to reach destinations in the network. Each node constructs distance-vectors and sends it to his neighbors. Routes are calculated by examining the vectors received and choosing the shortest path. DV protocols have problems such as slow convergence and creation of temporary forwarding loops. Several improvements were proposed to solve these problems [11]. RIP (Routing Information Protocol) [13] is nowadays the most used protocol in the Internet for intra-domain routing despite the growth of OSPF (Open Shortest-Path First) [14], a LS protocol.

In the LS model each router maintains a database with the complete network topology where each link is associated with a state. Link-state information is exchanged between routers. Since each node knows the entire topology, loops are easily avoided, the routing convergence is faster than with DV protocols, and source routing is easily implemented. However, LS protocols are inadequate for inter-domain routing because of the scalability and security issues involved. For inter-domain routing, protocols are currently being deployed based on variations of the DV algorithm, such as the path-vector approach [15]. Since in the LS model routers know the topology, it is easy to do per-flow routing [16]. On the other hand, for DV protocols the class-of-service approach is better suited.

## 3 Multi-constrained Routing

Route computation constrained by multiple QoS restrictions is complex [17,18]. The algorithms may treat the metrics in different ways. The first difference is related to the metric correlation. End-to-end transmission delay depends on the link capacity, the reserved bandwidth, the propagation delay, the link loss probability, and the traffic model employed. Ma and Steenkiste [17] proposed some algorithms based on dependent metrics but defining the metric relationship is a major problem.

Another way is to use metrics that are independent from each other. For instance, propagation delay is independent from the available bandwidth. In this case, the computational complexity of the algorithm is the main problem. Wang and Crowcroft [18] showed that for certain combinations of QoS requirements the problem is NP-complete.

This work considers independent metrics. The proposals found in the literature treat the metrics in one of three ways: individually, as an overall cost, or by means of heuristics. The next sections discuss these approaches.



### 3.1 Multiple Metrics

The complexity of the algorithm is function of the number of metrics treated and of their composition rules. The most usual requirements are delay, jitter, cost, packet loss probability, and bandwidth consumption. These metrics have different composition rules. If the metric of a path is the minimum value between all link metrics, the composition rule is concave. The additive and multiplicative cases are straightforward. Available bandwidth is concave. Delay, jitter, and cost are additive. The composition rule of loss probability is more complex. Let  $p$  be a path,  $lp(p)$  the path metric, and  $lp(i, j)$  the metric associated with the link between node  $i$  and node  $j$ . The composition rule for loss probability is given by

$$lp(p) = 1 - ((1 - lp(i, j)) \times (1 - lp(j, k)) \times \dots \times (1 - lp(q, r))). \quad (1)$$

However, if the transmission-success probability is used instead of the loss probability, the composition rule turns multiplicative.

Wang and Crowcroft [18] proved that finding a path subject to constraints on  $n$  additive and  $m$  multiplicative metrics is NP-complete if  $n + m \geq 2$ . Since bandwidth is concave, pruning infeasible links may treat it. Therefore, algorithms of reasonable complexity can only handle bandwidth and another metric.

### 3.2 Single Mixed Metric

Since algorithms based on a single metric are well known, a direct approach is to map the user's QoS requirements into a single measure and use it as metric. Given a path  $p$ , a mathematical function proportional to the bandwidth ( $B(p)$ ) and inversely proportional to the delay ( $D(p)$ ) and to the loss probability ( $L(p)$ ) can be defined (Equation 2). The path with the greater  $f(p)$  is the best route.

$$f(p) = \frac{B(p)}{D(p) \times L(p)}. \quad (2)$$

The single mixed metric has two problems. First, maximizing  $f(p)$  does not guarantee each of the QoS requirements. Second, the composition rule of this function is hard to define since the composition rules of its parameters are different. Nevertheless, the single mixed metric is used in many proposals [16,19]. This approach does not guarantee each QoS requirement individually, but adjusting the metric weights in the overall cost may increase the importance of a particular requirement.

### 3.3 Heuristics

There are different heuristic proposals for QoS routing [15,20,21]. The algorithm in [19,20] uses one metric to find a path, and then it tests if this path respects the other QoS requirements. If not, it uses another metric as the routing decision factor, and so on. When a feasible path is found the algorithm stops.

In general, these approaches may be based on two hypotheses. First, they may suppose centralized routing, i.e., the router knows the entire topology [19,20]. This is true for LS routing but not for DV routing. Second, some approaches

try to find a feasible path: this may lead to problems in hop-by-hop routing as route decisions made by all routers must be coherent at the risk of not being able to avoid loops. The use of feasible paths supposes some kind of connection-oriented behavior or source routing [15,21]. These hypotheses are not valid for hop-by-hop packet forwarding and DV routing. This paper proposes a routing algorithm adapted to hop-by-hop routing based on distance-vectors.

## 4 The SMM-DV Algorithm

In a companion paper, we proposed the Single Mixed Metric (SMM) routing [9]. This proposal is a simple source-routing algorithm that finds routes subject to three constraints: bandwidth, delay, and loss probability. The key idea is to execute a search in the network graph to eliminate all arcs that do not meet the bandwidth restriction and then to run a modified version of Dijkstra's algorithm. In this scheme, the source knows the network topology.

In this paper we propose and analyze the performance of the Single Mixed Metric - Distance Vector (SMM-DV) algorithm for hop-by-hop routing. SMM-DV uses a single mixed metric that combines delay and loss probability. Instead of loss probability, the algorithm uses the logarithm of transmission-success probability function (*slog*) to avoid complex composition rules. Furthermore, we assume that routes can not have more than 90% of loss probability, that is,  $0 < |slog| < 1$ . As *ms* unit is likely to be appropriate [22] to represent delay, we assume that delay is an integer. As a consequence, we have a simple single-metric representation of delay and loss, where the integer part is delay and the decimal part represents loss.

The usual approach in current hop-by-hop routing algorithms is to compute the best path to every destination. With a single metric, the best path is easily defined. With multiple metrics, however, the best path with optimal values for all parameters may not exist at all.

Actually, the objective of QoS routing is not to find the optimal path, but a feasible path. Nevertheless finding a feasible path is inadequate for distributed route computation based on distance-vectors. Permanent loops may be formed since routers do not know the entire topology. To avoid this, we adopted the approach of [18]. The shortest-widest path is loopless [18]. In fact, the "shortest" property avoids loops. This design decision guarantees the absence of loops at the expense of not finding some feasible paths.<sup>1</sup> In the algorithm that follows, the bandwidth precedes the delay metric. This centralized version of the algorithm is based on the Bellman-Ford's algorithm for distance-vector route computation.

Let  $G = (N, A)$  be a network with  $N$  nodes and  $A$  arcs. Each arc  $(i, j)$  is assigned two real numbers:  $b_{ij}$  is the available bandwidth and  $u_{i,j} = slog_{i,j} + d_{i,j}$  is the single mixed metric, where  $slog_{i,j}$  is the logarithmic transmission-success probability and  $d_{i,j}$  is the propagation delay.

<sup>1</sup> E.g., the shortest-widest route may not meet the delay constraint, while another route with less bandwidth (although still enough) meets the delay requirement.

When arc  $(i, j)$  is inexistent  $b_{ij} = 0$  and  $u_{ij} = \infty$ . Given a path  $p = (i, j, k, \dots, q, r)$ , the path width,  $w(p)$ , is  $\min[b_{ij}, b_{jk}, \dots, b_{qr}]$  and its length is  $l(p) = u_{ij} + u_{jk} + \dots + u_{qr}$ . Given three constants, **B** the minimum bandwidth, **D** the maximum delay, and **P** the maximum logarithmic transmission-success probability ( $\mathbf{U} = \mathbf{D} + \mathbf{P}$  is the maximum single mixed metric), the problem is to find a path,  $p$ , between  $i$  and  $r$  such that  $w(p) \geq \mathbf{B}$  and  $l(p) \leq \mathbf{U}$ : in our case, the shortest-widest path with three constraints.

Suppose node 1 is the source and  $h$  the number of arcs away from this node. Let  $B_i^{(h)}$  and  $U_i^{(h)}$  be the width and the length of the shortest-widest path from node 1 to node  $i$  within  $h$  hops. By convention,  $B_1^{(h)} = \infty$  and  $U_1^{(h)} = 0$  for all  $h$ . Adding a length checking, when multiple widest paths are found, produces the shortest-widest path algorithms. The SMM-DV algorithm is as follows:

**Step 1:**  $\forall i \neq 1 : h = 0$  and  $B_i^{(0)} = 0$   
**Step 2:**  $\forall i \neq 1$ , **find**  
 $K = \{k \mid w(1, \dots, k, i) = \max_{1 \leq j \leq N} [\min[B_j^{(h)}, b_{ji}]] \wedge w(1, \dots, k, i) \geq \mathbf{B}\}.$   
**Step 3:**  $\forall i \neq 1$ , **if**  $\#K > 1$ , **find**  $k \in K \mid$   
 $l(1, \dots, k, i) = \min_{1 \leq j \leq N} [U_j^{(h)} + u_{ji}] \wedge l(1, \dots, k, i) \leq \mathbf{U} \wedge slog(1, \dots, k, i) \leq \mathbf{P}.$   
**Step 4:**  $B_i^{(h+1)} = w(1, \dots, k, i)$  and  $U_i^{(h+1)} = l(1, \dots, k, i).$   
**Step 5:** **If**  $h \geq A$ , **the algorithm stops.**  
**If not,**  $h = h + 1$  and **go to step 2.**

Step 2 finds the maximum-bandwidth paths from node 1 to node  $i$  within  $h$  hops (and tests if the bandwidth is enough). When multiple widest paths are found, the minimum-length path is chosen (step 3). Step 3 also checks if the chosen path has a *slog* value smaller than **P**, where  $slog_{ij}$  is the decimal part of  $u_{ij}$ . Thus the loss probability is guaranteed without any additional search. Step 4 updates width and length of the path from node 1 to node  $i$ .

The optimal approach takes into account the metrics separately. This leads to a similar algorithm, where step 2 chooses the widest path, step 3 chooses the shortest path (minimum delay) and an additional step chooses the path with minimum loss probability. This additional search turns the approach unscalable. The optimal algorithm has two path lengths instead of one, so  $U_i^{(h)}$  is replaced by  $D_i^{(h)}$  and  $Slog_i^{(h)}$ . The optimal algorithm is as follows:

**Step 1:**  $\forall i \neq 1 : h = 0$  and  $B_i^{(0)} = 0$   
**Step 2:**  $\forall i \neq 1$ , **find**  
 $K = \{k \mid w(1, \dots, k, i) = \max_{1 \leq j \leq N} [\min[B_j^{(h)}, b_{ji}]] \wedge w(1, \dots, k, i) \geq \mathbf{B}\}.$   
**Step 3:**  $\forall i \neq 1$ , **if**  $\#K > 1$ , **find**  
 $Y = \{y \in K \mid d(1, \dots, y, i) = \min_{1 \leq j \leq N} [D_j^{(h)} + d_{ji}] \wedge d(1, \dots, y, i) \leq \mathbf{D}\}.$   
**Step 4:**  $\forall i \neq 1$ , **if**  $\#Y > 1$ , **find**  $y \in Y \mid$   
 $slog(1, \dots, y, i) = \min_{1 \leq j \leq N} [Slog_j^{(h)} + slog_{ji}] \wedge slog(1, \dots, y, i) \leq \mathbf{P}.$   
**Step 5:**  $B_i^{(h+1)} = w(1, \dots, y, i)$ ,  $D_i^{(h+1)} = d(1, \dots, y, i)$  and  $Slog_i^{(h+1)} = slog(1, \dots, y, i).$   
**Step 6:** **If**  $h \geq A$ , **the algorithm stops.**  
**If not,**  $h = h + 1$  and **go to step 2.**

The distributed version of these algorithms is simple. Each router periodically receives distance-vectors from its neighbors and chooses the best route among the possible options. SMM-DV takes advantage of the proposed single metric to find the route consuming fewer steps.

5 SMM-DV Simulation

We implemented the SMM-DV algorithm in NS-2[10]. NS-2 has a dynamic implementation of the Bellman-Ford routing algorithm that is the base of our work. Our implementation consists of two new routing modules and a modified link interface that admits three metrics: propagation delay, available bandwidth, and packet loss probability. The packet classifier takes into account the packet type and maps it onto three classes of service.

Table 1. Classes of Service.

CoS	QoS guarantees		
	Delay	Bandwidth	Loss probability
0	<b>10 ms</b>	1 Mbps	0.04
1	40 ms	<b>5 Mbps</b>	0.04
2	40 ms	1 Mbps	<b>0.01</b>

Each class of service (CoS) is characterized by QoS guarantees on bandwidth, delay, and loss probability. Within a specific class of service one requirement is more important than the others.<sup>2</sup> Table 1 presents the classes of service used in our implementation. The QoS values are not based on real applications but chosen to differentiate the classes of service. Each service has one strong guarantee: delay (CoS 0), bandwidth (CoS 1), or loss probability (CoS 2).

The optimal algorithm for CoS 1 is exactly the one presented in Section 4. Bandwidth is optimized, then delay and loss probability are minimized. The optimal algorithm for CoS 0 is obtained by inverting the order of steps 2 and 3 (so delay is first minimized); for CoS 2 step 4 precedes step 2 (so loss probability is first optimized).

The SMM-DV algorithm for CoS 1 is the one presented in Section 4, where bandwidth is first optimized, then the single mixed metric is minimized. CoS 0's algorithm first finds the paths that minimize the single mixed metric (they also minimize delay) and among these paths it chooses the one with maximum bandwidth. SMM-DV for CoS 2 finds the paths with minimum loss probability (the rational part of the single mixed metric) and afterwards the widest path among these. Delay is not minimized but the path is delay-constrained.

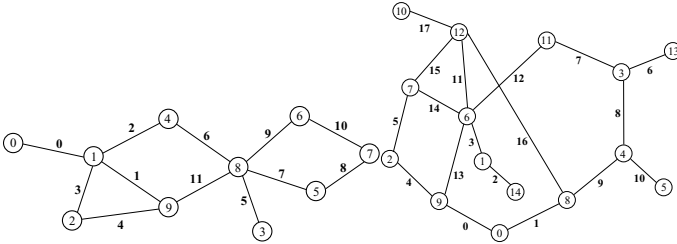
<sup>2</sup> For example, an application may have a strong requirement on the end-to-end delay but low bandwidth consumption (e.g., voice traffic). We can also imagine that for some applications the most important is to have high bandwidth available (e.g., video transmission), and real-time applications that demand high reliability (e.g., telemedicine) for which the loss probability must be minimized.

Our proposal was evaluated through the implementation of two protocols. The first one (3M) uses the optimal algorithm that searches the route using the metrics one by one. The second protocol uses the SMM-DV algorithm. SMM-DV does not find the route with optimal values for all three metrics, as does 3M, but finds a route constrained by the QoS requirements defined for each class of service. Therefore the routes may be different for some source-destination pairs.

The performance metric is the time complexity [23] expressed in steps: when a node receives routing update messages it recomputes its routes. A meter counts the number of loops needed by the routing algorithm to compute routes. This measure reflects the processing requirements of the algorithm. In the experiments link failures produced routing updates.

### 5.1 Topologies

Several topologies were produced by the GT-ITM topology generator [24]. The flat model was used. Nodes are placed on a plane with a grid: with a probability  $p$  a link is put between a pair of nodes; link delay is proportional to the distance between the nodes in this grid. The generator was modified to produce values for link delay, bandwidth, and loss probability. These values are used just for route computation. Physically, all links have the same bandwidth, delay, and loss probability. The objective is to measure the algorithm complexity. We generated topologies with 10, 15, 20, 25, 30, and 35 nodes. The parameter  $p$  was decreased for the 35 node topology due to limitations in NS.



**Fig. 1.** Two of the test topologies.

The model used does not represent well the Internet structure since it is purely random. Nevertheless it can model one routing domain: it is the approach used in [24], where Internet topology generation is studied. The model can give some clues on the issues involved in the deployment of inter-domain QoS routing. Despite its limitations, this model was useful in analyzing the complexity of the algorithm regarding routing dynamics.

### 5.2 Experiments

The objective of our experiments is to measure the processing requirements of QoS routing. This is done through the analysis of the trace files produced by the

meter mentioned before. Each simulation lasts for 2.5 seconds. During the first moments, all routers exchange messages that enable them to compute routes for every destination in the network. Then, at  $t = 1.0s$ , one link goes down, it goes up at  $t = 2.0s$ . The experiment is repeated for each link in the topology.

The Bellman-Ford algorithm has a problem of convergence: counting to infinity [11]. Protocol configuration limits “infinity” to a finite number (16 is usual with RIP [13]). For metrics other than the hop-count (integer) the definition of infinity is not so evident. Our first experiments presented problems due to too large values of “infinity”. With slow convergence, the nodes exchange a huge number of messages, this leads to congestion which causes routing messages losses, slower convergence and the production of more routing messages; this leads to more congestion and so on. The choices of the infinity value and of the metric granularity define the convergence speed. Therefore infinity was empirically chosen: 50 for delay, 0 for bandwidth and 0.05 for loss probability. With this approach it was possible to obtain simulations with topologies up to 25 nodes without routing-packet losses. With 30 nodes, there were some losses; with 35 nodes, as “infinity” became short, some source-destination pairs became unreachable. An option to setting various infinity values is to add the hop-count metric just for counting to infinity. Metric granularity influences also link-state QoS routing since it affects the amount of update messages produced [2,6].

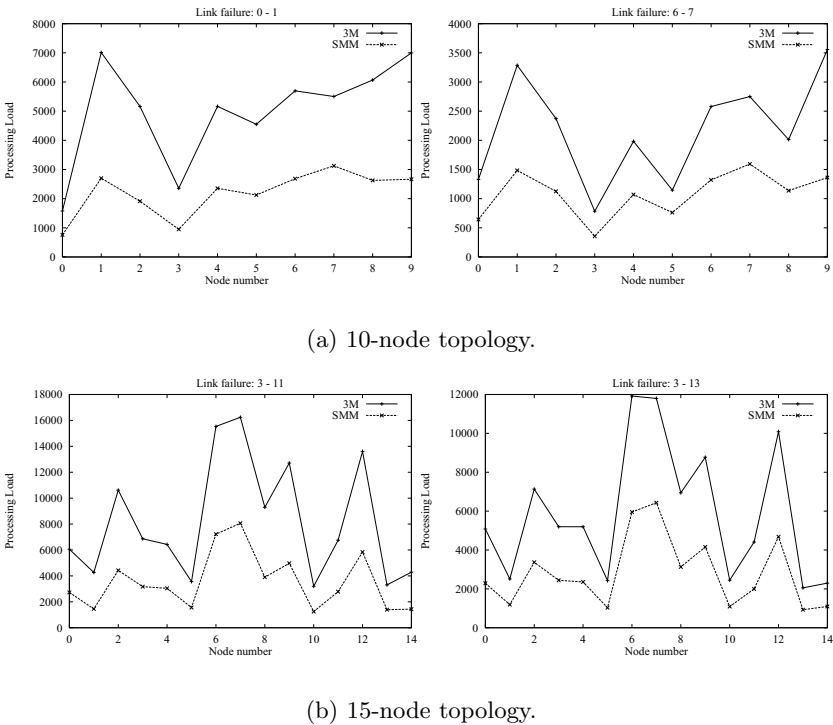
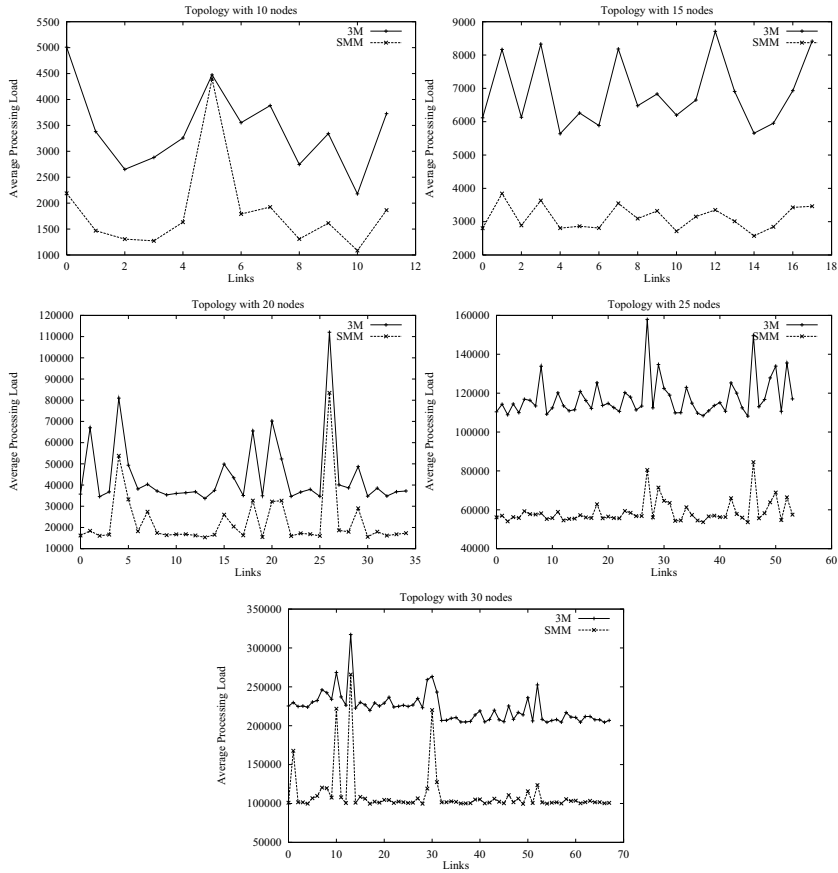


Fig. 2. Processing requirements at each node.

5.3 Analysis



**Fig. 3.** Global processing requirements.

Figure 2 shows experiments made for the ten and fifteen-node topologies presented in Figure 1. Each graph corresponds to a specific link failure. Nodes that have more neighbors experience a higher processing charge. Intuitively, these nodes have more options in choosing the next hop to a destination (and receive more routing messages than nodes with just one neighbor). This is the case of nodes **8**, **1**, and **9** in the 10-node topology and of nodes **6**, **7**, and **12** in the 15-node topology. On the other hand, nodes that have just one neighbor have the lightest load: **0** and **3** in Figure 1 and **5**, **10**, **13**, and **14**, as well as **1** (that has two neighbors, but one of them is **14**) in Figure 1.

The failure of link 0.1 produces a higher processing charge than that of link 6.7 in the 10-node topology (the graphs in Figure 2(a) have different scales). In the first case the network is disconnected: node **0** becomes unreachable forcing the other nodes to count to infinity.

Figure 3 has one graph per topology, each one present the average processing load (on all nodes in the topology) versus the link failure. The effect of different link failures is observed. The higher processing requirements are observed:

- when the topology is disconnected (links 0 and 5 in the 10-node topology, and links 3 and 17 in the 15-node topology);
- when the link failure adds to the graph diameter, slowing the routing convergence (links 7 and 9 in the 10-node topology and links 1, 7, and 12 in the 15-node topology).

SMM-DV may perform worse than 3M - see, e.g., when link 5 in the 10-node topology fails. Although SMM-DV discovers routes more quickly, it may present a slower convergence if 3M uses the delay metric for counting to infinity and SMM-DV uses the single mixed metric which has a finer granularity. Nevertheless, this is not likely to happen as our experiments showed. The global processing requirements increase with the topology size. The advantage of SMM-DV over 3M, the optimal algorithm, is as large as 50%.

## 6 Conclusion

QoS routing provides better QoS guarantees to applications and improve the network resource utilization. Nevertheless QoS routing is a costly mechanism. Improved scalability can be achieved by class-of-service based routing.

Scalability and security issues arise in the use of link-state protocols for QoS routing. Storing the complete network topology does not scale and knowing the entire topology is not always the case. Therefore a distance-vector like protocol is more adapted for QoS routing in some cases.

SMM-DV is an algorithm for distance-vector QoS routing that computes routes per class of service taking into account three metrics. SMM-DV is as complex as the two-metric algorithm proposed in [18]. This paper compared the processing requirements of SMM-DV with that of an optimal algorithm, 3M. Simulations used randomly generated topologies. The average performance gain of SMM-DV is as large as 50%. We also observed that the network topology influences the processing requirements of DV QoS routing. Nodes that have more neighbors experience bigger processing charge. On the other hand, the link failure is more critical when the network is disconnected and when the network diameter is increased. Therefore DV QoS routing takes advantage of highly connected networks.

## References

1. E. Crawley, R. Nair, B. Rajagopalan, and H. Sandick, *RFC2386: A Framework for QoS-Based Routing*. Network Working Group, Aug. 1998.
2. G. Apostolopoulos, R. Guerin, S. Kamat, and S. K. Tripathi, "Quality of service based routing: A performance perspective," in *ACM SIGCOMM'98*, pp. 17–28, Sept. 1998.



3. N. Traft-Plotkin, B. Bellur, and R. Ogier, "Quality-of-Service routing using maximally disjoint paths," in *IEEE/IFIP IWQoS'99*, June 1999.
4. D. H. Lorenz and A. Orda, "Qos routing in networks with uncertain parameters," in *IEEE INFOCOM'98*, 1998.
5. V. Fayet, D. A. Khotimsky, and T. Przygienda, "Hop-by-hop routing with node-dependent topology information," in *IEEE INFOCOM'99*, Mar. 1999.
6. A. Shaikh, J. Rexford, and K. G. Shin, "Evaluating the overheads of source-directed quality-of-service routing," in *International Conference on Network Protocols*, 1998.
7. R. Braden, D. Clark, and S. Shenker, *RFC1633: Integrated Services in the Internet Architecture: An Overview*. Network Working Group, June 1994.
8. S. Blake, D. Black, M. Carlson, E. Davies, Z. Wang, and W. Weiss, *RFC2475: An Architecture for Differentiated Services*. Network Working Group, Dec. 1998.
9. L. H. M. K. Costa and O. C. M. B. Duarte, "A scalable QoS-based routing mechanism for supporting multimedia applications," in *IEEE International Conference on Multimedia Computing and Systems - ICMCS'99*, vol. 2, pp. 346–351, June 1999.
10. K. Fall and K. Varadhan, *NS notes and documentation*. University of California at Berkeley/Lawrence Berkeley National Laboratory, Mar. 1999. Available at <http://www-mash.cs.berkeley.edu/ns>.
11. C. Huitema, *Routing in the Internet*. Prentice Hall, 1996.
12. B. Davie and J. Gibson, "Enabling explicit routing in IP networks," in *IEEE GLOBECOM'98 - Internet Mini-Conference*, pp. 125–130, 1998.
13. G. Malkin, *RFC1721: RIP Version 2 Protocol Analysis*. Network Working Group, Nov. 1994.
14. J. Moy, *RFC1583: OSPF version 2*. Network Working Group, Mar. 1994.
15. R. Vogel, R. G. Herrtwich, W. Kalfa, H. Wittig, and L. C. Wolf, "QoS-based routing of multimedia streams in computer networks," *IEEE Journal on Selected Areas in Communications*, vol. 14, no. 7, pp. 1235–1244, Sept. 1996.
16. R. Guerin, A. Orda, and D. Williams, "QoS routing mechanisms and OSPF extensions," in *IEEE GLOBECOM'97*, Nov. 1997.
17. Q. Ma and P. Steenkiste, "Quality of service routing for traffic with performance guarantees," in *IFIP Fifth International Workshop on Quality of Service*, (New York), pp. 115–126, May 1997.
18. Z. Wang and J. Crowcroft, "Quality of service routing for supporting multimedia applications," *IEEE Journal on Selected Areas in Communications*, vol. 14, no. 7, pp. 1228–1234, Sept. 1996.
19. L. L. H. Andrew and A. A. N. A. Kusuma, "Generalised analysis of a QoS-aware routing algorithm," in *IEEE GLOBECOM'98*, pp. 118–123, 1998.
20. W. C. Lee, M. G. Hluchyj, and P. A. Humblet, "Routing subject to quality of service constraints in integrated communication networks," *IEEE Network Magazine*, vol. 9, no. 4, pp. 46–55, July-Aug. 1995.
21. H. F. Salama, D. S. Reeves, and Y. Viniotis, "A distributed algorithm for delay-constrained unicast routing," in *IEEE INFOCOM'97*, 1997.
22. A. Fei, G. Pei, R. Liu, and L. Zhang, "Measurements on delay and hop-count of the Internet," in *IEEE GLOBECOM'98 - Internet Mini-Conference*, 1998.
23. J. J. Garcia-Luna-Aceves and J. Behrens, "Distributed, scalable routing based on link-state vectors," *IEEE Journal on Selected Areas in Communications*, vol. 13, no. 8, pp. 1383–1395, Oct. 1995.
24. E. W. Zegura, K. Calvert, and S. Bhattacharjee, "How to model an internetwork," in *IEEE INFOCOM'96*, vol. 2, pp. 594–602, Mar. 1996.

# On Shortest Path Problems with “Non-Markovian” Link Contribution to Path Lengths

Arunabha Sen<sup>1</sup>, K. Selçuk Candan<sup>1</sup>, Afonso Ferreira<sup>2</sup>, and Bruno Beauquier<sup>2</sup>,  
and Stephane Perennes<sup>2</sup>

<sup>1</sup> Department of Computer Science and Engineering  
Arizona State University  
Tempe 85287, AZ, USA  
{asen, candan}@asu.edu

<sup>2</sup> SLOOP, CNRS - INRIA - UNSA  
2004 Rote des Lucioles, BP 93  
F-06902 Sophia-Antipollis, FRANCE  
{Afonso.Ferreira, Beauquier, Stephane.Perennes}@sophia.inria.fr

**Abstract.** In this paper we introduce a new class of shortest path problems, where the contribution of a link to the path length computation depends not only on the weight of that link but also on the weights of the links already traversed. This class of problems may be viewed as “non-Markovian”. We consider a specific problem that belong to this class, which is encountered in the multimedia data transmission domain. We consider this problem under different conditions and develop algorithms. The shortest path problem in multimedia data transmission environment can be solved in  $O(n^2)$  or  $O(n^3)$  computational time.

## 1 Introduction

Path problems have been extensively studied by many researchers of Computer Science and Operations Research because of its applications in many problems in these domains. In most of these problems, one or more weights are associated with a link representing, among other things, the *cost*, *delay* or the *reliability* of that link. The objective most often is to find a least weighted path between a specified source-destination pair. In almost all the path problems studied so far (and discussed in the literature), the contribution of a link to the path length computation depends only on the weight of that link and is independent of the weights of the links already traversed. This condition is similar to a Markov chain where the next state is dependent only on the current state and is independent of the past states. In this paper, we introduce a new variant of the path problem. In this variant, the contribution of a link to the path length computation depends not only on the weight of that link but also on the weights of the links already traversed. This class of problems may be viewed as “non-Markovian” as the contribution of a link towards the path length depends on the current link as well as the links traversed in the past on the path from the source to the destination.

As an example, we consider a specific problem that belongs to this class. This problem is encountered in the multimedia data transmission domain. We consider this problems under different conditions and develop appropriate algorithms. The shortest path problem in multimedia data transmission environment can be solved in  $O(n^2)$  or  $O(n^3)$  computational time. We also provide mathematical programming solutions for this problem.

## 2 Prior Work

Shortest path problems are among the most widely studied problems in Computer Science and Operations Research. Because of its wide applications in many diverse domains, these problems have been studied for at least the last forty years [1,4,5]. In the earliest version of the shortest path problem [1,4,5], a weight is associated with each link of the network and the path length is computed by summing up the weights of the links belonging to the path. In a generalization of this version of the problem, multiple weights are associated with each link of the network. If there are  $m$  different weights associated with each link of the network, there are  $m$  different path lengths associated with each path. The  $i$ -th path length, ( $1 \leq i \leq m$ ), is obtained by summing up the  $i$ -th weights of the links belonging to the path. This version of the shortest path problem is known as *multicriteria shortest path problem* or *constrained shortest path problem* and is fairly extensively studied in [6,8,12]. In view of the attention that the *Quality of Service* issues have received in the networking community in recent years, study of this version of the shortest path problem has become increasingly important [14].

Another version of the shortest path problem that has received considerable attention is the one where the weights associated with the links of the network are allowed to change with time. Both centralized as well as distributed algorithms for the shortest path in this scenario have been developed under various waiting constraints in [9,10]. In yet another version of the problem, each link,  $e$ , of the network has two weights, transit time  $b(e, u)$  and cost  $c(e, u)$ , where  $u$  is the departure time at the starting node of the link. In this version, the problem is to find the least cost path such that the total traversal time is below some prespecified threshold value  $T$ . A dynamic programming algorithm for the shortest path problem with time windows and additional linear costs on node service start times is presented in [7]. In [11] the authors consider a version of the problem termed as the *quickest path problem*, where the objective is to transfer a specified amount of data from the source to the destination with minimum transmission time. The transmission time in this problem is dependent on both the capacities and the traversal times of the links in the network. The shortest path problem in multimedia data transmission environment is discussed in [3].

### 3 Problem Formulation

In the classical path problem, each edge  $e_i \in E$  of the graph  $G = (V, E)$  has a weight  $w_i$  associated with it and if there is a path  $P$  from the node  $v_0$  to  $v_k$

$$v_0 \xrightarrow{w_1} v_1 \xrightarrow{w_2} v_2 \xrightarrow{w_3} \dots \xrightarrow{w_k} v_k$$

then the *path length* or the *distance* between the nodes  $v_0$  and  $v_k$  is given by

$$PL(v_0, v_k) = w_1 + w_2 + \dots + w_k$$

This model is valid as long as the weights on the links represents the *cost* or the *delay* associated with the link. However, if the weight represents the *reliability* or the *bandwidth* associated with the link, then *addition* of the link weights on the path is not meaningful. In case, the weights represent the reliability, the calculation once again becomes meaningful if the addition operator is replaced by a *multiplication* operator. In case, the weight represents the bandwidth, the calculation becomes meaningful if a *minimum* operator replaces the addition operator. Thus a generalization of the path length will be

$$PL(v_0, v_k) = w_1 \oplus w_2 \oplus w_3 \oplus \dots \oplus w_k$$

where  $\oplus$  is a suitable operator for the particular application. In [14], the authors consider three different types of operators and call them *additive*, *multiplicative*, and *concave metrics* respectively.

At the next level of generalization, the path length computation is based on not the link weight itself but a function of the link weight. In this case the path length is given by

$$PL(v_0, v_k) = f(w_1) \oplus f(w_2) \oplus f(w_3) \oplus \dots \oplus f(w_k)$$

where  $f(w_i)$  can be any function of the link weight  $w_i$ , appropriate for the particular application.

At the next higher level of generalization each link has multiple weights associated with it. This model realistically captures the data transmission environment where the *Quality of Service (QoS)* issues are of paramount importance. The various weights associated with a link may represent among other things, the *delay*, the *cost*, the *jitter*, the *cell loss rate* etc. In this case the path length computation is carried out in one of the following two ways:

*Case I:* In this case each path has multiple *path lengths* associated with it. If  $(w_{i,1}, w_{i,2}, \dots, w_{i,m})$  are  $m$  different link weights associated with the link  $e_i$ , then there are  $m$  different path lengths,  $[PL_1(v_0, v_k), \dots, PL_m(v_0, v_k)]$ , associated with a *path* between a given source node  $v_0$  and a given destination node  $v_k$ , where

$$PL_i(v_0, v_k) = f(w_{1,i}) \oplus f(w_{2,i}) \oplus \dots \oplus f(w_{k,i})$$

This class of problems is known as the *multicriteria optimization problems* and is studied in [6,8,12,14].

*Case II:* In this case each path has a single path length associated with it:

$$PL(v_0, v_k) = f(w_{1,1}, \dots, w_{1,m}) \oplus f(w_{2,1}, \dots, w_{2,m}) \oplus \dots \oplus f(w_{k,1}, \dots, w_{k,m})$$

It may be noted that this formulation gives rise to a *single criterion optimization problem* as opposed to the *multiple criteria optimization problem* in Case I.

Both Case I and Case II of the previous level can be further generalized at the next higher level. As in the previous case, each link has multiple weights associated with them. In this level of generalization, the contribution of a link in the path length computation depends not only on the weights associated with that link but also on the weights of the links already traversed. In this case the path length,  $[PL_1(v_0, v_k), \dots, PL_m(v_0, v_k)]$ , for Case I is such that

$$PL_i(v_0, v_k) = f(w_{1,i}) \oplus \dots \oplus f(w_{1,i}, \dots, w_{k,i}).$$

At this level of generalization, the path length for Case II is

$$PL(v_0, v_k) = f(w_{1,1}, \dots, w_{1,m}) \oplus f(w_{1,1}, \dots, w_{1,m}, w_{2,1}, \dots, w_{2,m}) \oplus \dots \oplus f(w_{1,1}, \dots, w_{1,m}, \dots, w_{k,1}, \dots, w_{k,m}).$$

We say that the edges in this category have “non-Markovian” link contributions.

## 4 Path Problem in Multimedia Data Transmission

The example path problem discussed in this paper belongs to this last category. This problem is based on a multimedia data transmission model we recently presented in [3]. The model allows an active network to perform certain operations to the data at the network nodes. These operations, such as format conversions for distributed multimedia collaborations and lossy/lossless compressions may change the sizes and qualities of multimedia object being transmitted. In this paper, we use a subset of this model, where the quality is not taken into account for path selection.

In the variant of the path problem for the multimedia data transmission, each edge  $e_i$  has two weights,  $\delta_i$  and  $s_i$  associated with it,  $\delta_i \geq 0$  and  $s_i \geq 0$ . These two weights are referred to as (i) the *per unit delay factor* and (ii) the *size factor* respectively. If  $P$  is a path from the node  $v_0$  to  $v_k$ ,

$$v_0 \xrightarrow{\delta_1, s_1} v_1 \xrightarrow{\delta_2, s_2} v_2 \xrightarrow{\delta_3, s_3} \dots \xrightarrow{\delta_k, s_k} v_k.$$

then the *path length* or the *total delay* between the nodes  $v_0$  and  $v_k$  denoted  $PL(v_0, v_k)$  is given by

$$\begin{aligned} PL(v_0, v_k) &= \delta_1 + s_1\delta_2 + s_1s_2\delta_3 + \dots + s_1 \dots s_{k-1}\delta_k \\ &= \sum_{i=1}^k \delta_i \prod_{j=1}^{i-1} s_j \quad \text{with} \quad \prod_{j=1}^0 s_j = 1. \end{aligned}$$

It is clear that the path length in this case fits into the most general case discussed in the previous paragraph with  $m = 2, w_{i,1} = \delta_i, w_{i,2} = s_i$  and  $f(w_{1,1}, w_{1,2}, w_{2,1}, w_{2,2}, \dots, w_{i,1}, w_{i,2}) = s_1 s_2 \dots s_{i-1} \delta_i$ , for all  $i, 1 \leq i \leq k$ , and  $s_0 = 1$ .

The physical significance of the parameters  $\delta_i$  and  $s_i$  are as follows: The transmission delay is clearly proportional to the size of the multimedia data file being transmitted. Therefore we consider the *per unit delay factor*  $\delta_i$  and to compute the total delay, we multiply  $\delta_i$  with the size of the file being transmitted. As a multimedia data file travels through different nodes in a network on its journey from the source to the destination, it passes through some transformation algorithms. As a result, the size of the multimedia data file may change. The size factor  $s_i$  captures this aspect of multimedia data transmission. We remark, however, that the total delay remains proportional to the amount of data transmitted along the path. Therefore, the expression for the path length (or total delay) stated above is given for transmitting *one unit of data* from the source to the destination.

#### 4.1 Why Is This Problem Different?

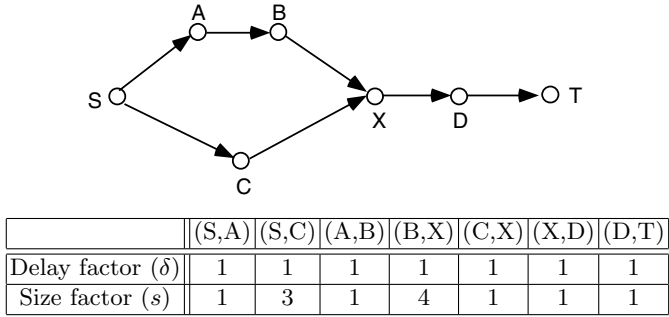
The length of a path  $P(v_0, v_k) : v_0 \rightarrow v_1 \rightarrow v_2 \rightarrow \dots \rightarrow v_k$ , in the multimedia data transmission problem, is given by  $PL(v_0, v_k) = \delta_1 + s_1 \delta_2 + s_1 s_2 \delta_3 + \dots + s_1 \dots s_{k-1} \delta_k$ . The traditional shortest path algorithms such as the *Dijkstra's algorithm* make the observation that “*subpaths of shortest paths are shortest paths*” and exploits it to develop the shortest path algorithm.

In other words, to get to the destination from the source using the shortest path, the intermediate nodes must be visited using the shortest path from the source to the intermediate nodes. This is true because, the *path length* in this case is computed as the sum of weights on the links that make up the path. In case of multimedia data transmission problem, where *the path length is not computed as the sum of the links weights, this is no longer true*. This is demonstrated with an example shown in Figure 1. The  $\delta_i$  and  $s_i$  values associated with the links of this graph is also given in Figure 1.

With this data set the length of the path,  $S \rightarrow C \rightarrow X \rightarrow D \rightarrow T$ , is  $\delta_{S,C} + s_{S,C} \delta_{C,X} + s_{S,C} s_{C,X} \delta_{X,D} + s_{S,C} s_{C,X} s_{X,D} \delta_{D,T} = 1 + 3.1 + 3.1.1 + 3.1.1.1 = 1 + 3 + 3 + 3 = 10$  whereas the length of the path  $S \rightarrow A \rightarrow B \rightarrow X \rightarrow D \rightarrow T$  is  $\delta_{S,A} + s_{S,A} \delta_{A,B} + s_{S,A} s_{A,B} \delta_{B,X} + s_{S,A} s_{A,B} s_{B,X} \delta_{X,D} + s_{S,A} s_{A,B} s_{B,X} s_{X,D} \delta_{D,T} = 1 + 1.1 + 1.1.1 + 1.1.4.1 + 1.1.4.1.1 = 1 + 1 + 1 + 4 + 4 = 11$ . Thus the path  $S \rightarrow C \rightarrow X \rightarrow D \rightarrow T$  is shorter than the path  $S \rightarrow A \rightarrow B \rightarrow X \rightarrow D \rightarrow T$  in the example. However, in this example the length of the path  $S \rightarrow C \rightarrow X$  is  $1 + 3.1 = 4$ , which is *greater* than the length of the path  $S \rightarrow A \rightarrow B \rightarrow X$ ,  $1 + 1.1 + 1.1.1 = 3$ .

On the other hand, the path length function in the multimedia data transmission problem has an interesting property and this property is utilized to establish the following lemma.

**Lemma 1.** *Given a weighted directed graph  $G = (V, E)$  with weight functions  $(\delta_i, s_i)$ , associated with each link  $e_i$ , ( $e_i \in E, 1 \leq i \leq |E|$ ), and the length of a*



**Fig. 1.** An Example Graph for MMD Transmission and the Associated Delay and Size Factors

path  $P(v_0, v_k) : v_0 \rightarrow v_1 \rightarrow \dots \rightarrow v_k$  is computed as  $PL(v_0, v_k) = \delta_1 + s_1\delta_2 + s_1s_2\delta_3 + \dots + s_1 \dots s_{k-1}\delta_k$ . Let  $P(v_0, v_k) : v_0 \rightarrow v_1 \rightarrow \dots \rightarrow v_k$  be a shortest path from vertex  $v_0$  to vertex  $v_k$  and for any  $i$ ,  $1 \leq i \leq k-1$ , let  $P(v_i, v_k) : v_i \rightarrow v_{i+1} \rightarrow \dots \rightarrow v_k$  be a subpath of  $P$  from vertex  $v_i$  to vertex  $v_k$ . Then  $P(v_i, v_k)$  is a shortest path from  $v_i$  to  $v_k$ ,  $1 \leq i \leq k$ .

The proof of the lemmas and theorems in this paper are omitted for space considerations. The proofs may be found in [13].

**4.2 Path Problem in Multimedia Data Transmission with No Reduction in Size**

In this subsection, we consider a special case where the size factor,  $s_i$ , associated with a link  $e_i$  is greater than or equal to unity for all the links. This implies that the data size will never reduce from its original size while passing through a link. The more general case where the size factor,  $s_i$ , does not have any such restriction (i.e.,  $s_i$  is allowed to be less than unity) will be considered in the next subsection.

Beacuse of lemma 3 and the fact  $s_i \geq 1, \delta_i \geq 0$ , we can apply a modified version of Dijkstra’s algorithm to solve the shortest path problem in the multimedia data transmission environment. The traditional version of the algorithm starts from the source node and computes the shortest path to other nodes until it finds the shortest path to the destination. In this modified version, we start from the destination node and compute the shortest path from other nodes to the destination nodes until it finds the shortest path from the source to the destination node. The algorithm is given in Figure 2.

**Theorem 1.** If  $\forall i, j$  the delay factor  $\delta(i, j) \geq 0$  and the size factor  $s(i, j) \geq 1$  then the above algorithm correctly computes the shortest path from any node  $i, 1 \leq i \leq n-1$  to the destination node  $n$ .

**Theorem 2.** The complexity of the algorithm is  $O(n^2)$ .

**Shortest Path Algorithm for Multimedia Data Transmission Environment**

*Input:* The directed graph  $G = (V, E)$ , ( $V = \{1, 2, \dots, n\}$ ), two  $n \times n$  matrices  $\delta$  and  $s$ , the  $(i, j)$ -th entry of the matrices stores the delay factor  $\delta$  and the size factor  $s$  of the link from the node  $i$  to node  $j$ . If there is no link from the node  $i$  to  $j$ , both  $\delta_{i,j}$  and  $s_{i,j}$  is taken to be  $\infty$ . Without any loss of generality, we assume that the node 1 is the source node and node  $n$  is the destination node.

*Output:* Array  $D(1, \dots, n)$ , that stores the shortest path length from node  $i$  to the destination node  $n$  for all  $i$ ,  $1 \leq i \leq n$ .

*Comments:* The algorithm starts from the destination node and in each iteration finds the shortest path from a node  $i$  in the graph to the destination node  $n$ ,  $1 \leq i \leq n - 1$ .

**begin**

$C := \{1, 2, \dots, n - 1\};$

**for**  $i := n - 1$  **downto** 1 **do**

$D[i] := \delta[i, n]$

**repeat**  $n - 2$  **times**

**begin**

$v := i \in C$  such that  $D[i]$  has the minimum value;

$C := C \setminus \{v\};$

**for each**  $w \in C$  **do**

$D[w] := \min(D[w], \delta[w, v] + s[w, v]D[v]);$

**end**

**end**

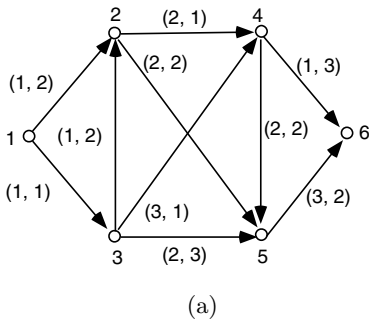
**Fig. 2.** Shortest Path Algorithm for Multimedia Data Transmission Environment

An example graph and the corresponding result of the execution of the algorithm on the graph is shown in Figure 3 (the source node is 1 and the destination is 6). The shortest path length from the node 1 to node 6 is 5 and the path is  $v_1 \rightarrow v_3 \rightarrow v_4 \rightarrow v_6$ .

It is well known that if the path length is measured as the sum of the weights on the links, Dijkstra's algorithm fails to compute the shortest path between the source-destination nodes, in case some of the link weights are *negative*. For exactly the same reason, our modified version of the Dijkstra's algorithm fails to compute the shortest path if  $s_{i,j} < 1$ . An example of the case where the above algorithm fails to compute the shortest path is shown in Figure 4. The  $\delta_i$  and  $s_i$  values associated with the links of this graph is also given in Figure 4 (a). The result of the execution of the modified Dijkstra algorithm on this graph is shown in Figure 4.

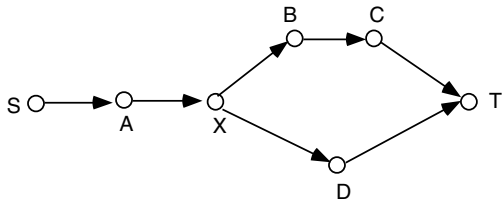
At the termination of the algorithm, the shortest path length between the source node  $S$  and the destination node  $T$  is given as 6 and the path is  $S \rightarrow A \rightarrow X \rightarrow D \rightarrow T$  ( $\delta_{S,A} + s_{S,A}\delta_{A,X} + s_{S,A}s_{A,X}\delta_{X,D} + s_{S,A}s_{A,X}s_{X,D}\delta_{D,T} = 1 + 1.2 + 1.1.2 + 1.1.1.1 = 6$ ). However, this result is *incorrect* because the length of the path  $S \rightarrow A \rightarrow X \rightarrow B \rightarrow C \rightarrow T$  is  $\delta_{S,A} + s_{S,A}\delta_{A,X} + s_{S,A}s_{A,X}\delta_{X,B} +$





Iteration	Nodes of the Graph					
	1	2	3	4	5	6
1	$\infty$	$\infty$	$\infty$	1*	3	0
2	$\infty$	3	4	1	3*	0
3	$\infty$	3*	4	1	3	0
4	7	3	4*	1	3	0
5	5*	3	4	1	3	0

**Fig. 3.** (a) An Example Graph for MMD transmission and (b) the Corresponding Shortest Path Computation



	(S,A)	(A,X)	(X,B)	(X,D)	(B,X)	(C,T)	(D,T)
$\delta$	1	2	1	2	2	2	1
$s$	1	1	0.25	1	1	1	1

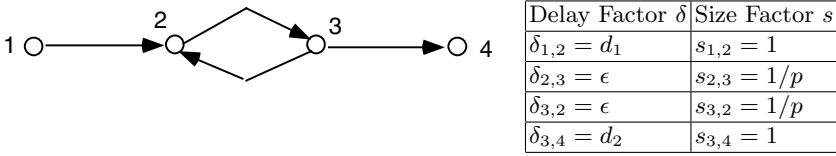
Iteration	Nodes of the Graph						
	S	A	X	B	C	D	T
1	$\infty$	$\infty$	$\infty$	$\infty$	2	1*	0
2	$\infty$	$\infty$	3	$\infty$	2*	1	0
3	$\infty$	$\infty$	3*	4	2	1	0
4	$\infty$	5	3	4*	2	1	0
5	$\infty$	5*	3		2	1	0
6	6*	5	3	4	2	1	o

**Fig. 4.** (a) An Example Graph for MMD Transmission and (b) the Corresponding Shortest Path Computation

$s_{S,A}s_{A,X}s_{X,B}\delta_{B,C} + s_{S,A}s_{A,X}s_{X,B}s_{B,C}\delta_{C,T} = 1 + 1.2 + 1.1.1 + 1.1.(0.25).2 + 1.1.(0.25).1.2 = 5$ . In this case, the algorithm computes the shortest path length incorrectly, because one of the size factors,  $s_{X,B} < 1$  ( $s_{X,B} = 0.25$ ).

**4.3 Path Problem in Multimedia Data Transmission with Reduction in Size**

It was mentioned in the previous section that in this path problem if some size factor  $s_i < 1$ , it has the same effect as a negative weighted link in a traditional shortest path problem. The example given earlier, shows that our version of the Dijkstra’s algorithm fails to correctly compute the shortest path from the source to the destination in this situation. In the traditional shortest path problem, where the path length is computed as the sum of the weights on the links of a path, there is a notion of a *negative weighted cycle*. A cycle is referred to as a *negative weighted cycle* if the sum of the weights on the links making up the cycle is a negative number. The multimedia data transmission problem that is



**Fig. 5.** An Example Graph with Negative Weighted Cycle and the Associated Delay and Size Factors

presently under consideration, both the weights (the delay factor  $\delta_i$  and size factor  $s_i$ ) associated with a link  $e_i$ , are non-negative. However, in this problem path length is computed in a different way. In this problem also we have a notion of a negative weighted cycle. The implication of such a negative weighted cycle is that the data size decreases every time it goes around such a cycle in such a way that the total delay is also reduced every time the data goes around such a cycle. An example of such a phenomenon is given next.

Consider the graph in Figure 5 with the nodes 1 and 4 being the source and the destination respectively. The nodes 2 and 3 form a loop, as shown in the figure. Now, consider the path  $1 \rightarrow 2 \rightarrow 3 \rightarrow 4$ . This is a *no loop* path between the nodes 1 and 4. The path length of this path is  $d_1 + 1 \cdot \epsilon + 1 \cdot (1/p) \cdot d_2 = d_1 + \epsilon + d_2/p$ . The length of the path from 1 to 4, if it passes through the loop  $1, 2, \dots, k$  times is

$$d_1 + (1 + 1/p + \dots + 1/p^{2k})\epsilon + d_2/p^{2k+1}.$$

Thus the path length increases by  $((p+1)/p^{2k+2})\epsilon$  and decreases by  $((p^2 - 1)/p^{2k+3})d_2$  if the number of times the path goes through the loop increases from  $k$  to  $k+1$ . If  $d_2$  is much larger than  $\epsilon$ , then the total decrease is much larger than the total increase and as a result if the path goes through the loop one more time, the path length decreases. The situation is similar to the *negative weighted cycle* problem in the traditional shortest path length.

In the path problem for multimedia data transmission environment, we can compute the shortest path between a specified source-destination pair, even with “negative” weights (i.e., with size factor  $s_i < 1$ ) on the links, as long as there is no negative weighted cycles in the graph. We use a modified version of the Bellman-Ford algorithm for this purpose.

Just like the traditional Bellman-Ford algorithm, we find the shortest path lengths subject to the constraint that paths contain at most one link, then relax the condition on the length of the path and find the shortest path length subject to the constraint that paths contain at most two links and so on. Using the same terminology and notations as in [2] we call the shortest path that uses at most  $h$  links as the *shortest* ( $\leq h$ ) *path*.

Suppose that  $D^h_i$  denotes the shortest ( $\leq h$ ) path length from node  $i$  to the destination node  $n$ , ( $1 \leq i \leq n-1$ ).  $D^h_n = 0$  for all  $h$ . The algorithm is given in Figure 6.

**Shortest Path Algorithm for Multimedia Data Transmission Environment**

*Input:* The directed graph  $G = (V, E)$ , ( $V = \{1, 2, \dots, n\}$ ), two  $n \times n$  matrices  $\delta$  and  $s$ , the  $(i, j)$ -th entry of the matrices stores the delay factor  $\delta$  and the size factor  $s$  of the link from the node  $i$  to node  $j$ . If there is no link from the node  $i$  to  $j$ , both  $\delta_{i,j}$  and  $s_{i,j}$  is taken to be  $\infty$ . Without any loss of generality, we assume that the node 1 is the source node and node  $n$  is the destination node.

*Output:* The shortest path length from every node in the graph to the destination node  $n$ .

*Comments:* The algorithm starts from the destination node and in each iteration finds the shortest ( $\leq h$ ) path from a node  $i$  in the graph to the destination node  $n$ ,  $1 \leq i, h \leq n - 1$ .

**begin**

**for**  $i := 1$  **to**  $n-1$  **do**

$D^0_i := \infty$ ;

**for**  $i := 1$  **to**  $n-1$  **do**

$D^1_i := \delta(i, n)$ ;

**for**  $h := 1$  **to**  $n-2$  **do**

**for**  $i := 1$  **to**  $n-1$  **do**

$D^{h+1}_i := \min_{1 \leq j \leq n-1} [s(i, j)D^h_j + \delta(i, j)]$ ;

**end**

**Fig. 6.** Shortest Path Algorithm for Multimedia Data Transmission Environment

**Theorem 3.** *If the graph  $G = (V, E)$  does not contain any negative weighted cycle, then the above algorithm correctly computes the shortest path length from any node  $i$ ,  $1 \leq i \leq n - 1$  to the destination node  $n$ , even when some of the size factors  $s_i$  associated with a link  $e_i$  is less than 1.*

**Theorem 4.** *The complexity of the algorithm is  $O(n^3)$ .*

An example of the result of the execution of the algorithm on the graph in Figure 4 is shown in Table 1.

#### 4.4 Mathematical Programming Solution to the Path Problem in Multimedia Data Transmission

In this subsection we show that the shortest path problem for the multimedia data transmission problem can also be solved using mathematical programming techniques.

Given a graph  $G = (V, E)$  with weights  $\delta_i$  and  $s_i$  associated with each link  $e_i \in E$  and two specified vertices  $s$  and  $t$ , the problem is to find a shortest (or the least weighted) path from  $s$  to  $t$ .

In the mathematical programming formulation of the problem, we associate a binary indicator variable  $x_{i,j}$  with each link  $i, j$  of the directed graph  $G = (V, E)$ .

**Table 1.** Shortest Path Computation for the Graph in Figure 4 using modified Bellman-Ford Algorithm

Iteration Number	Nodes of the Graph						
	S	A	X	B	C	D	T
1	$\infty$	$\infty$	$\infty$	$\infty$	2	1	0
2	$\infty$	$\infty$	3	4	2	1	0
3	$\infty$	$\infty$	2	4	2	1	0
4	$\infty$	4	2	4	2	1	0
5	5	4	2	4	2	1	0
6	5	4	2	4	2	1	0

By assigning a zero or a one to the variable  $x_{i,j}$  the solution indicates whether or not the link  $i, j$  is a part of the shortest path from the source to the destination. We also introduce two other variables  $y_{i,j}$  and  $z_{i,j}$ , ( $1 \leq i, j \leq |V|$ ). By assigning a value to the variable  $y_{i,j}$  (resp.  $z_{i,j}$ ), the solution indicates how much data is entering (resp. leaving) the link  $(i, j)$ .

*Minimize  $\sum_{(i,j) \in E} \delta_{i,j} y_{i,j}$  subject to the following constraints:*

1.  $\sum_{\{j|(i,j) \in E\}} x_{i,j} - \sum_{\{j|(j,i) \in E\}} x_{j,i} = r_i, \forall i \in V$   $r_i = 1$  if  $i$  is the source node,  $r_i = -1$  if  $i$  is the destination node and  $r_i = 0$  if  $i$  is any other node.
2.  $y_{s,j} = x_{s,j} \forall j \in V - \{s\}$  and  $(s, j) \in E$ , ( $s$  is the source node)
3.  $\sum_{\{j|(i,j) \in E\}} y_{i,j} - \sum_{\{j|(j,i) \in E\}} z_{j,i} = 0, \forall i \in V - \{s, t\}$
4.  $z_{i,j} = s_{i,j} y_{i,j}, \forall (i, j) \in E$
5.  $z_{i,j} \leq K x_{i,j}, \forall (i, j) \in E$  where  $K$  is a large constant.

The first constraint establishes a path from the source to the destination. As data passes through a link  $(i, j)$ , its size changes by a factor  $s_{i,j}$ . This is ensured by the constraint (iv). The delay keeps accumulating as the data file passes through various links on its journey from the source to the destination. This aspect is captured by the constraint (iii). The purpose of constraint (v) is to ensure that the variable  $z_{i,j}$  does not have a non-zero value when  $x_{i,j} = 0$ , i.e., when the link  $(i, j)$  is not a part of the path from the source to the destination. The contribution of the delay factor  $\delta_{i,j}$  associated with the link  $(i, j)$  is taken into account in the objective function of the formulation.

### 5 Conclusion

In this paper we introduced a new class of the shortest path problems. In this class of path problems, the contribution of a link towards the path length depends not only on the weight of the link itself but also on the weights of all the links traversed before traversing the link under consideration. We considered a specific path problem that belong to this new class. This problem is encountered in multimedia data transmission domain. Exploiting an interesting property of the multimedia transmission path problem, we could develop low order polynomial

time algorithms for this problem. Additionally, we could solve this problem using mathematical programming techniques. Other path problems that belong to this class are currently under investigation.

## References

1. R. Bellman, "On a Routing Problem", *Quarterly of Applied Mathematics*, vol. 16, no. 1, pp. 87-90, 1958.
2. D. Bertsekas and R. Gallager, *Data Networks*, Prentice Hall, 1987.
3. K.S. Candan and Y. Yang, "Least-Cost High-Quality Object Retrieval for Distributed Multimedia Collaborations", *IEEE Multimedia Computing and Systems Conference*, pp. 649-654, Florence, Italy, June 1999.
4. E.W. Dijkstra, "A Note on Two Problems in Connection with Graphs", *Numerische Mathematik*, vol. 1, pp. 269-271, 1959.
5. L.E. Ford and S.M. Johnson, "A Tournament Problem", *The American Mathematical Monthly*, vol. 66, pp. 387-389, 1959.
6. G.Y. Handler and I. Zang, "A Dual Algorithm for the Constrained Shortest Path Problem", *Networks*, vol. 10, pp. 293-310, 1980.
7. I. Ioachim et al., "A Dynamic Programming Algorithm for the Shortest Path Problem with Time Windows and Linear Node Costs", *Networks*, vol. 31, pp. 193-204, 1998.
8. J.M. Jaffe, "Algorithms for Finding Paths with Multiple Constraints", *Networks*, vol. 14, pp. 95-116, 1984.
9. A. Orda and R. Rom, "Shortest-Path and Minimum-Delay Algorithms in Networks with Time-Dependent Edge-Lengths", *Journal of the Association for Computing Machinery*, vol. 37, no. 3, pp. 605-625, 1990.
10. A. Orda and R. Rom, "Minimum Weight Paths with Time-Dependent Networks", *Networks*, vol. 21, pp. 295-319, 1991.
11. J.B. Rosen, S. Z. Sun and G.L. Xue, "Algorithms for the Quickest Path Problem and the Enumeration of Quickest Paths", *Computers and Operation Research*, vol. 18, no. 6, pp. 579-584, 1991.
12. H.F. Salama, D.S. Reeves and Y. Viniotis, "A Distributed Algorithm for Delay-Constrained Unicast Routing", *IEEE INFOCOM'97*, pp. 1c.2.1-1c.2.8, 1997.
13. A. Sen, K. S. Candan, A. Ferreira, B. Beauquier, S. Perennes, "On Shortest Path Problems with "non-Markovian" Link Contribution to Path Lengths", Tech. Report-00-002, Dept. of Computer Science and Engineering, Arizona State University.
14. Z. Wang and J. Crowcroft, "Quality-of-Service Routing for Supporting Multimedia Applications", *IEEE Journal on Selected Areas of Communications*, vol. 14, no. 7, pp. 1228-1234, 1996.

# Adaptive QoS Platform in Multimedia Networks

Mahmoud Sherif<sup>1</sup>, Ibrahim Habib<sup>1</sup>, Mahmoud Naghshineh<sup>2</sup>, and Parviz Kermani<sup>2</sup>

<sup>1</sup> CUNY Graduate School and  
Department of Electrical Engineering  
The City College of New York, USA  
{sherif, eeiwh}@ee-mail.engr.cuny.cuny.edu

<sup>2</sup> IBM Research Division,  
T.J. Watson Research Center  
Yorktown, USA

**Abstract.** In an adaptive multimedia environment, each of the multimedia substreams (i.e. video, audio and data) has its own distinct quality of service (QoS) requirements (e.g. cell loss rate, delay, jitter, etc.). These requirements constitute a certain QoS level. In contrast to the static approach, each substream declares a preset range of acceptable QoS levels (e.g., high, medium, low) instead of just a single one. This range of QoS levels is defined in a user-defined profile (UDP). In this paper, we suggest a channel borrowing algorithm based on an adaptive QoS platform. In a channel borrowing algorithm, an acceptor cell that has used all its nominal channels can borrow free channels from its neighboring cells (candidate donors) to accommodate new calls. In our suggested algorithm, an acceptor cell can borrow from any neighboring (donor) cell as long as this donor cell has some channels available after satisfying a minimum QoS (minQ) level defined in the UDP. A donor cell assigning QoS levels (to calls under its coverage) higher than the minQ levels defined in the UDP will declare those channels as available for borrowing by other acceptor cells. When a channel is borrowed, several other cells are prohibited from using it due to channel locking. The proposed channel borrowing algorithm differs in the way a free channel is selected from a donor cell to be borrowed by an acceptor cell. The criteria for choosing the free channel include not only the number of free channels but also the QoS levels in the donor cell. The criteria is also extended to include the effect of channel locking on the number of free channels and the QoS levels on the locked cells.

## 1 Introduction

In wireless multimedia networks, admission control is required to reserve resources in advance for calls requiring guaranteed services. In the case of a multimedia call, each of its substreams (i.e. video, audio and data) has its own distinct quality of service (QoS) requirements (e.g. cell loss rate, delay, jitter, etc.). The network attempts to deliver the required QoS by allocating an appropriate amount of resources (e.g., bandwidth, buffers). The negotiated QoS requirements constitute a certain QoS level

that remains fixed during the call (static allocation approach). Accordingly, the corresponding allocated resources also remain unchanged. In a previous paper [1], we presented and analyzed an adaptive allocation of resources algorithm based on genetic algorithms. In such an adaptive networking environment, calls can be admitted to the system even if the available bandwidth is not sufficient to satisfy their highest QoS guarantees. To accomplish this, the proposed algorithm will try to degrade the QoS levels of the existing calls in order to free some bandwidth. Each call is considered to have a pre-defined minimum QoS level (minQ) and a maximum QoS level (maxQ) that are defined in a user-defined profile (UDP). The (minQ) level corresponds to the minimum set of QoS requirements that the application is willing to tolerate. On the other hand, the (maxQ) level corresponds to the best QoS requirements a call can obtain whenever the bandwidth is available. In the case of a multimedia call, each of its substreams (i.e. video, audio and data) will have its own QoS level ranging from high to medium to low. For example, one user might be willing to tolerate a low video quality, but requires high audio quality and high speed data service. Such user will be granted an amount of bandwidth that corresponds to its declared minQ level. Whenever there is available bandwidth, the user might obtain an extra amount of bandwidth up to its declared maxQ level. It should be noted that the UDP approach also defines different “grades of service” at different costs that the users can subscribe to. For example, a network provider may choose to offer two grades of services with different cost structures. A subscriber to a premium service will pay high dollar for the call but will be allocated a UDP with the uppermost range of quality levels, whereas a subscriber to an economy service will have a UDP range of qualities that is significantly lower than the premium one.

In fig.1 it is shown that when a call is being admitted to the network, it presents its traffic parameters and its declared UDP to the admission controller/scheduler. The admission controller/scheduler will then calculate the required bandwidth according to the traffic parameters presented to it. The calculated bandwidth and the declared UDP are then saved in a database in order to be used by the proposed optimization algorithm. Fig.2 shows an example of the UDP where a call (i) is presenting both the maxQ and the minQ levels to the admission controller/scheduler. All QoS levels between the declared maxQ and minQ are allowed to be assigned to call i. In this example, the maxQ level corresponds to the highest video quality, the highest audio quality and the highest speed for data service. The minQ level corresponds to medium video quality, low audio quality and low speed for data service. Any QoS level between these declared levels are allowed to be allocated to this call i.

In this paper, we suggest a channel borrowing algorithm based on the adaptive allocation of resources algorithm described in [1]. In a channel borrowing algorithm, an acceptor cell that has used all its nominal channels can borrow free channels from its neighboring cells (candidate donors) to accommodate new calls. In our suggested algorithm, an acceptor cell can borrow from any neighboring (donor) cell as long as this donor cell has some channels available after satisfying the minimum QoS (minQ) level defined in the (UDP). A donor cell assigning QoS levels (to calls under its coverage) higher than the minQ levels defined in the UDP will declare those channels as available for borrowing by other acceptor cells. A channel can be borrowed by a cell if

the borrowed channel does not interfere with existing calls. Furthermore, when a channel is borrowed, several other cells are prohibited from using it. This is called channel locking. The number of such cells depends on the cell layout and the type of initial allocation of channels to cells. In contrast to static borrowing, channel borrowing strategies deal with short-term allocation of borrowed channels to cells. Once a call is completed, the borrowed channels are returned to its nominal cell. The proposed channel borrowing algorithm differs in the way a free channel is selected from a donor cell to be borrowed by an acceptor cell. In our suggested algorithm, the criteria for choosing the free channel include not only the number of free channels but also the QoS levels in the donor cell. The criteria is also extended to include the effect of channel locking on the number of free channels and the QoS levels on the locked cells.

Throughout the description of the suggested channel borrowing algorithm, we are going to consider the hexagonal planar layout of the cells. A cell cluster is a group of identical cells in which all of the available channels (frequencies) are evenly distributed. The most widely used plan is the N=7 cell cluster [2]-[4] where the number of available channels are distributed evenly among 7 cells, which then repeats itself over and over according to Fig. 3. We are going to consider N=7 throughout the discussion in this paper.

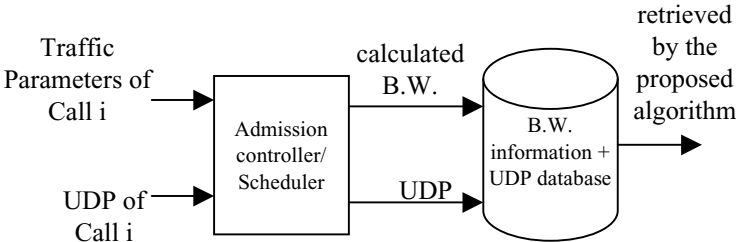


Fig. 1. Database used by the algorithm

	Video	Audio	Data
<b>maxQ</b>	<b>High</b>	<b>High</b>	<b>High</b>
	High	High	Medium
	High	High	Low
	...	...	...
	...	...	...
	Medium	Medium	Low
<b>minQ</b>	<b>Medium</b>	<b>Low</b>	<b>Low</b>

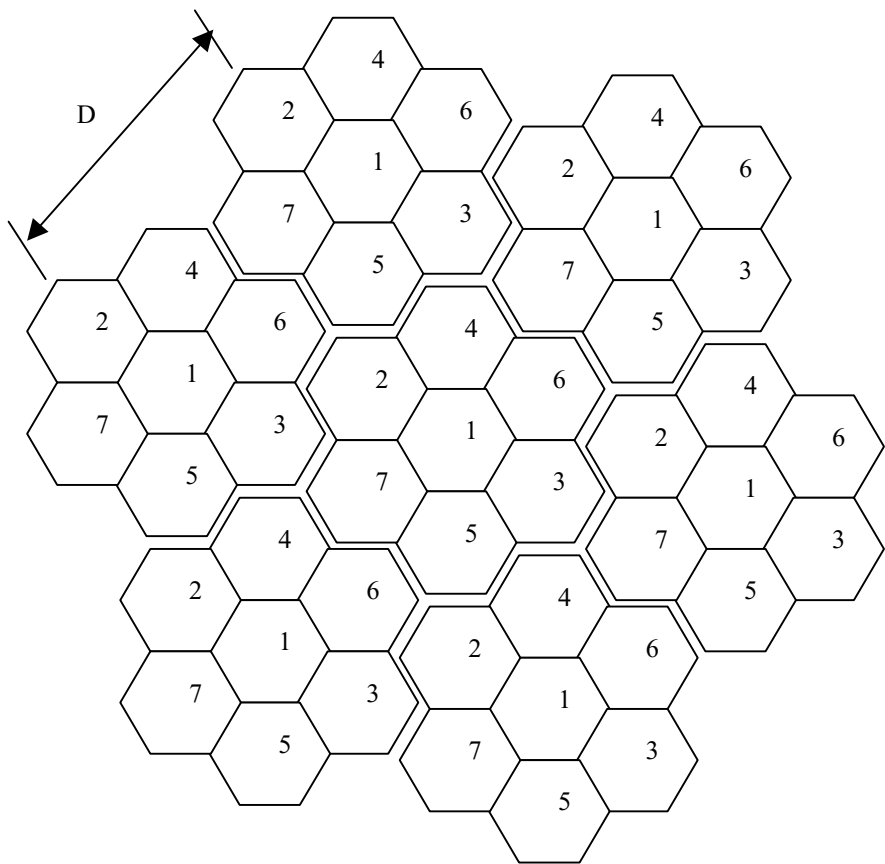
Fig. 2. Example of a User-Defined Profile (UDP)

2 Suggested Channel Borrowing Algorithm

As opposed to the *Borrow from the richest* [5] algorithm, the suggested algorithm takes into account the effect of channel locking when choosing a candidate channel for



borrowing. Furthermore, the suggested algorithm takes into account not only the number of available channels but also the average QoS level of each candidate cell. This allows the algorithm to try to maximize the average QoS level of the calls existing in the system in addition to minimizing the call blocking probability.



**Fig. 3.** A seven-cell reuse plan

Fig. 4 shows the block diagram of the adaptive allocation of resources algorithm when the channel borrowing algorithm is added. As shown, the algorithm consists of three main modules: genetic algorithm modules I and II in addition to the channel borrowing module III. The reason to use modules I and II is to divide the problem into smaller easier to handle sub-tasks since the search space is too large for a single GA. Module I assigns fair bandwidth allocations to the existing calls; whereas module II maximizes the capacity utilization by assigning any available bandwidth (left over from module I allocations) to the existing calls. Eventually, the system will not force a call to drop unless there is not enough bandwidth to satisfy its minQ level. Module III is added as an important enhancement layer to allow for channel borrowing to take place in such an adaptive QoS environment.

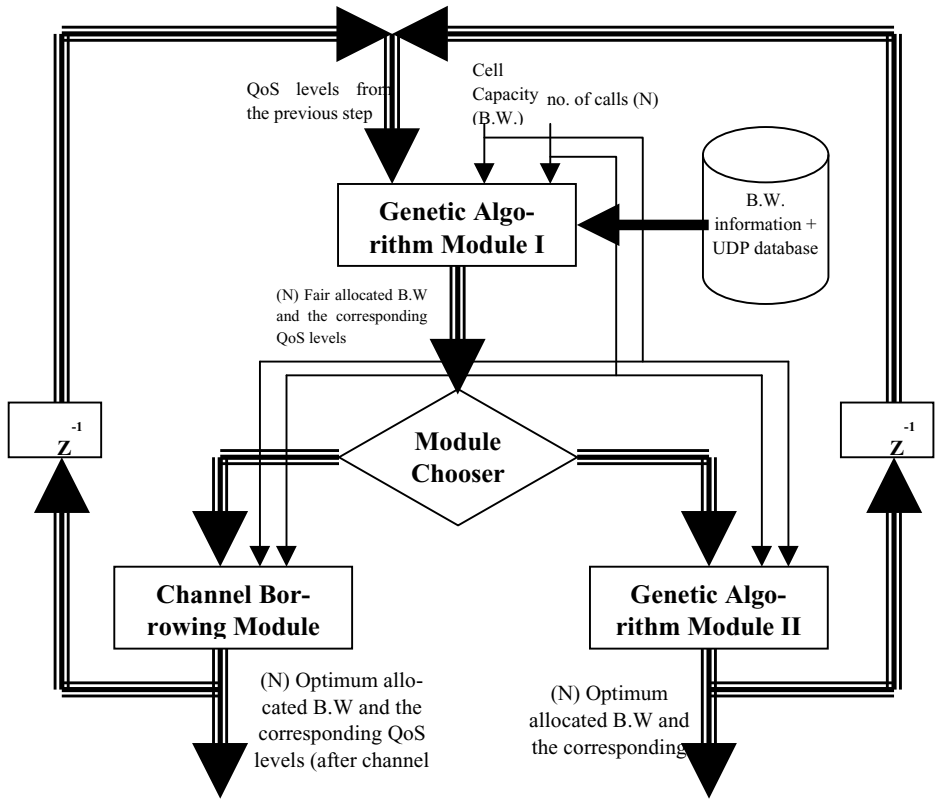


Fig. 4. Block Diagram of the Proposed Adaptive QoS Platform

Module I is triggered whenever a call arrival or departure takes place. The inputs to module I include: (1) the capacity of the system (physical cell capacity), (2) the number of calls ( $N$ ) after the arrival/departure and (3) the bandwidth information for these calls and their preset UDPs (which are fetched from the database). Furthermore, QoS levels from the previous allocation are also fed back to module I.

Module I searches for the QoS levels that correspond to the bandwidth allocation closest to the "fair" bandwidth. The evaluation of the optimization function is based upon the following rules:

$$B_{fair} = \frac{C_{tot}}{N} \quad (1)$$

Where

$B_{fair}$  is the fair bandwidth Allocation, and

$C_{tot}$  is the total physical cell capacity.

In Eq.2, the calculation of the fair bandwidth is applicable to calls of similar classes (i.e., having the same type of multimedia substreams)

For each of the existing call, the following equations are applied:

$$B_{\text{modl}}(i) \in B_i(Q_i) \quad (2)$$

$$B_{\text{modl}}(i) = \text{minimum}(\text{absolute}(B_{\text{fair}} - B_i(Q_i))) \quad (3)$$

$$B_{\text{modl}}(i) \leq B_{\text{fair}} \quad (4)$$

Where

$i$  is the call number,

$B_{\text{modl}}(i)$  is the bandwidth requirement corresponding to the output QoS level from module I,

$B_i(Q_i)$  is the bandwidth requirement corresponding to the QoS level  $Q_i$ , and

$B_{\text{fair}}$  is the fair bandwidth.

Once the fair allocations are determined by module I, if any of the assigned QoS levels in module I is less than the corresponding minimum QoS level  $\text{min}Q$ , then this  $\text{min}Q$  level is assigned to the corresponding call. Then this data is sent to the module chooser for further processing. Fig. 5 shows the module chooser. It starts by calculating the total bandwidth needed by all existing calls if granted the QoS levels assigned by module I. If the total bandwidth exceeds the cell capacity, then module III is triggered to try to borrow some bandwidth from the neighboring cell. Otherwise, module II is triggered to try to take advantage of any bandwidth left over from module I.

Module II, on the other hand, is designed to take advantage of any available bandwidth left over from module I. It tries to maximize the link capacity utilization and thus maximizing the QoS levels for the calls

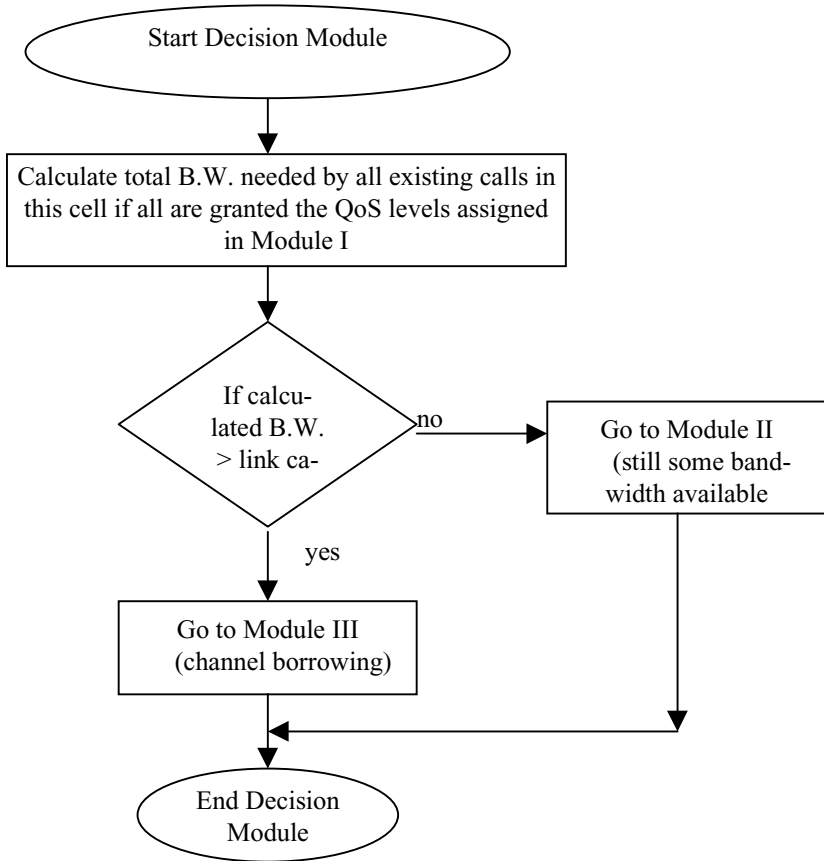
In the following section, we are going to describe module III (channel borrowing algorithm) in more details.

## 2.1 Module III

Throughout the description of module III, we are going to consider the cell layout shown in Fig. 6. The cell layout shown is a seven-cell cluster surrounded by a second tier of 11 cells. The total number of cells in this two-tier layout is 19 cells. The cell reuse plan is  $N=7$ . Each color in Fig. 6 represents a distinctive set of channel frequencies.

The reason for choosing this specific layout is that module III is going to consider the first two tiers of the neighboring cells only. Cell A is the acceptor cell. It is surrounded by two tiers of cells. Cell A can borrow from any of its first tier neighbors (cell 1, cell 2, ..., cell 6). Furthermore, it can borrow channels from one and only one neighbor (no multiple donor cells are allowed). Borrowing any number of channels from any of these 6 cells will cause these channels to be locked in another two cells. To illustrate the concept of channel locking, let us assume that cell A is going to choose to borrow a number of free channels from cell 2. This will cause these borrowed channels to be locked in both cells 14 and 17. For this cell layout ( $N=7$ ), the number of cells affected by channel locking is always 2 cells. We are going to denote

these 2 cells as *affected\_cell1* and *affected\_cell2*. In the suggested channel borrowing algorithm, each cell is required to keep track of the following parameters: (1) The number of available free channels if the existing calls are assigned their minQ levels, (2) the number of available free channels in *affected\_cell1* if the existing calls are assigned their minQ levels and (3) the number of available free channels in *affected\_cell2* if the existing calls are assigned their minQ levels. The information regarding the available number of channels in the *affected\_cell* can be collected either periodically or on a need basis (a message sent from the cell to the *affected\_cell* requesting immediate information regarding the number of available free channels).

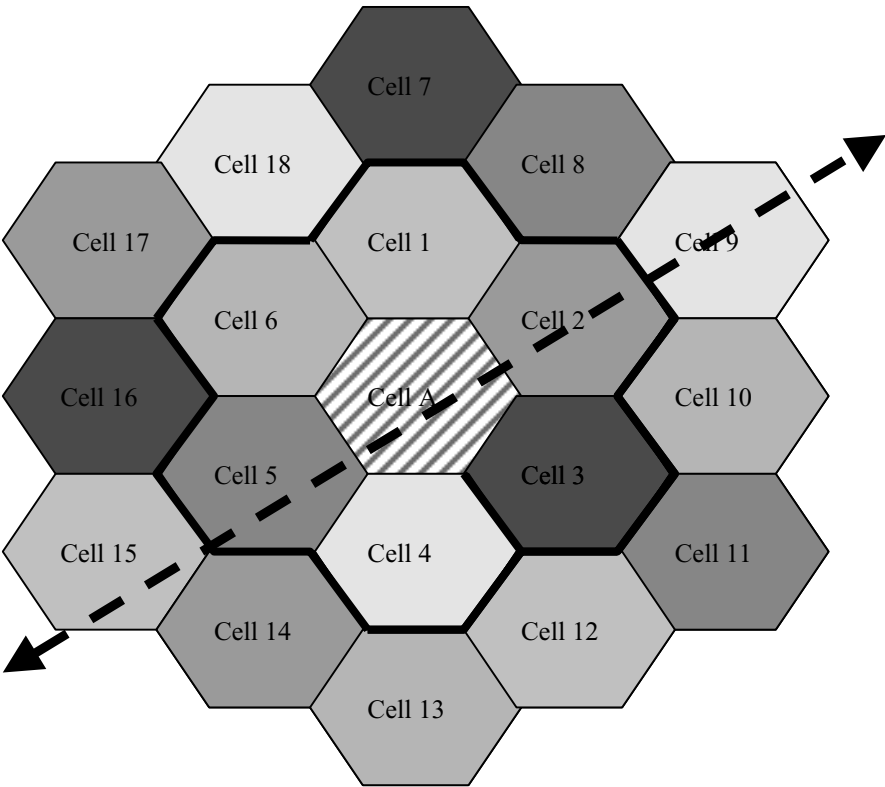


**Fig. 5.** Module Chooser

Assuming that Cell A (the acceptor cell) needs  $N_b$  number of channels to borrow. The number of channels needed ( $N_b$ ) should be calculated from the following equation:

$$N_b = (B_{\text{tot}} - C_{\text{tot}}) / Ch_{\text{size}} \quad (5)$$

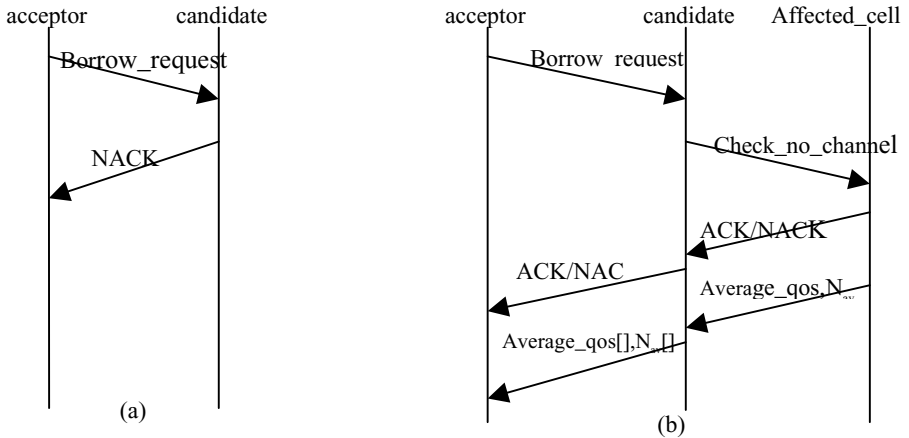
Where  
 $B_{tot}$  is the total bandwidth needed by existing calls if assigned their minQ level,  
 $C_{tot}$  is the total physical cell capacity and  $Ch_{size}$  is the size of each channel.



**Fig. 6.** System Under Study

Cell A will then send a *borrow\_request* message to each of its first tier neighbor cells. Each of the neighboring cells receiving the *borrow\_request* message will calculate the number of available channels ( $N_{av}$ ) if its calls are granted their minQ level and compare it to the requested number of channels ( $N_b$ ). If  $N_{av} > N_b$ , then the cell will send back a negative acknowledge (NACK) message informing the acceptor of the denial of its request. This is shown in Fig. 7 (a). Otherwise, it will send a message to each of the affected cells (due to channel locking) requesting information regarding the number of available channels ( $N_{av}(\text{affected\_cell})$ ). The information regarding the available channels does not have to be gathered when the *borrow\_request* message is received. As mentioned before, this information can be gathered periodically and kept ready for any *borrow\_request* message. If the returned values  $N_{av}(\text{affected\_cell}) > N_b$

then this means that the requested number of channels is available for borrowing and that the locked channels (in the affected\_cells) are also available. An acknowledgement (ACK) message is then sent back to the acceptor cell. Otherwise, a NACK message is sent back to the acceptor cell denoting that the locked channels will cause some calls to be dropped in the affected\_cell. The following routine can be executed by the neighboring cells upon the receipt of a *borrow\_request* message.



**Fig. 7.** Flow of messages between acceptor, candidate and affected cells (a)  $N_{av} < N_b$ , (b)  $N_{av} > N_b$

Fig. 7 (b) shows the flow of the messages if there are enough available channels in the neighboring (candidate) cell ( $N_{av} > N_b$ ). If the affected cells return an ACK message, this implies that the locked channels due to the borrowing process will not cause any calls to be dropped in these cells. This is due to the fact that when the candidate cell sends a *check\_no\_channel* message to any of the affected cells, it will calculate the number of available channels after assigning the minQ levels to the calls existing in this cell. Therefore, immediately afterwards, the candidate cell send an ACK message to the acceptor cell informing it that it can fulfill the *borrow\_request* message.

The candidate cell will then wait for each of the affected cells to send a message carrying information regarding the average QoS (*average\_qos*) levels of the existing calls and the number of channels that would be available if the borrowing process is executed ( $N_{av}$ ). The *average\_qos* is calculated assuming the borrowing process has been executed. This will give an indication of how the borrowing process affects the QoS levels of the existing calls. In the meantime, the candidate cell will calculate the average QoS levels of its existing calls based on the same criteria. Notice that the candidate cell can calculate the average QoS level while the other affected cells are doing the same thing in parallel. Once all the information is available for the candidate cell, it will send it in a message to the acceptor cell.

Once all messages from all the neighboring (candidate) cells are received, the acceptor cell will start processing the information searching for the best candidate cell to become the donor. First, all cells sending back NACK messages are removed from the

candidate list. The information of each of the remaining cells in the candidate list include: (1) a vector of average QoS levels of the candidate cell, affected cell 1 and affected cell 2 ( $average\_qos[]$ ), and (2) a vector of available number of channels ( $N_{av}$ ).

The acceptor cell will use the following equation to calculate the cost of borrowing the requested number of channels from each cell. The cost value of each cell in the current candidate list is denoted by  $borrow\_cost$ .

$$borrow\_cost = \left( \frac{\min(N_{av})}{\max\_channels} - \frac{\min(average\_qos)}{64} \right) \quad (6)$$

Where  $\max\_channels$  is the maximum number of channels in each of the cells. Note that the value of  $average\_qos$  ranges from 1 to 64 (1 being the best QoS level, and 64 being the least). Therefore, the value of  $borrow\_cost$  ranges from -1 to 1. The higher the  $borrow\_cost$  value, the better is the candidate cell. Therefore, the acceptor cell will choose the one with the highest  $borrow\_cost$  value.

### 3 Simulation Results

The system under study is shown in Fig. 6. The system consists of an acceptor cell (cell A) surrounded by two tiers of cells. The cell layout is a seven-cell cluster system ( $N=7$ ). Each cell has a physical capacity of 60,000 ATM cells(packets)/sec. This is equivalent to 25 Mbps. For a 30 Kbps channels, each cell is assigned 848 channels ( $\max\_no\_channels=848$ ). The dotted line represents a street with high call traffic load. Along the street are cells 15, 5, A, 2 and 9. In our simulation, the traffic load in these cells is characterized by a high traffic load. As shown in the cell layout, each of the following 6 sets represent a group of cells assigned the same set of channel frequencies: (1) cells {1,12,15}, (2) cells {2,14,17}, (3) cells {3,7,16}, (4) cells {4,9,18}, (5) cells {5,8,11} and (6) cells {6,10,13}. Therefore, the  $no\_of\_affected\_cells=2$  (i.e. when one cell is a donor cell, 2 other cells are affected due to channel locking).

The traffic load is multimedia traffic. In our simulation, the video component of the multimedia call was obtained using the MPEG-1 coded movie *Star Wars* as an example of high activity video traffic. Voice and Data substreams were generated according to an exponential distribution. The MPEG-1 coded movie generates a frame sequence of *I B B P B B P B B P B B*. There are 24 frames per second [6]. This set of data is described in [7]. Each call has an average duration of 5 minutes of movie time. Therefore the movie was actually divided into 24 different calls. To generate three QoS levels for the video substream, we adopted the algorithm described in chapter 2 where the B frames are dropped to obtain a lesser QoS level, then both the B and P frames are dropped to obtain the least video quality. Whereas, the high, medium and low QoS levels for audio were generated by varying the average bit rate of the audio source from 32 to 16 to 8 Kbps, respectively. Similarly, QoS levels for data were generated by varying the data rate from 64 to 32 to 16 Kbps.

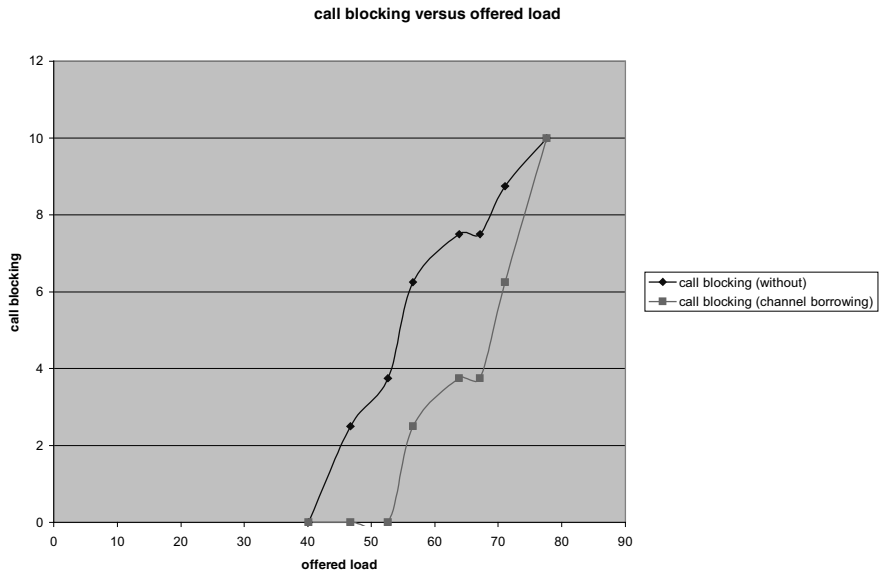


Fig. 8. . Call Blocking versus offered load

Fig. 8 shows the percentage call blocking rate of the system under study versus the percentage offered load. Using the suggested channel borrowing algorithm reduced the call blocking rate of the system. This takes place under an offered load between 40% to 80%. Below 40%, there is no need for using the channel borrowing algorithm as there are enough channel in the system to satisfy at least the minQ levels of the exist- ing calls.

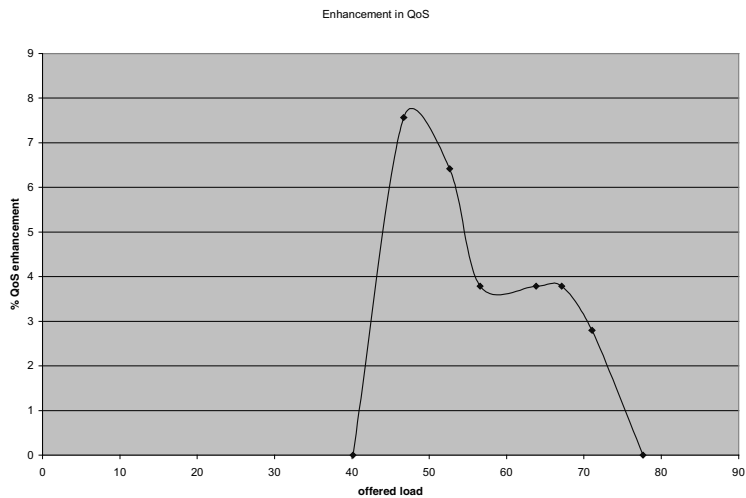


Fig. 9. QoS enhancement versus offered load



Fig. 9 shows the enhancement in the QoS offered versus the offered load. This curve shows another contribution of the suggested algorithm. Due to the adaptive QoS platform of the suggested system, the suggested channel borrowing algorithm is also able to enhance the QoS level of existing calls. The enhancement starts when the offered load is at 40%. It reaches a peak value of 7.8% at 45% offered load. It then starts to decrease reaching 0% at 78% offered load.

## References

1. Sherif, M.R., Habib, I.W., Naghshineh, M., Kermani, P.: Adaptive Allocation of Resources and Call Admission Control in Wireless ATM Using Genetic Algorithms. *IEEE Journal Select. Areas Comm.* February, 2000.
2. Michalewicz, Z.: *Genetic Algorithms + Data Structures = Evolution Programs*. 3rd edn. Springer-Verlag, Berlin Heidelberg New York (1996).
3. Farouque, S.: *Cellular Mobile Systems Engineering*. Artech House, Boston, 1996.
4. Holtzman, J., Goodman, D.: *Wireless Communications, Future Directions*. Kluwer Academic Publishers, 1993.
4. Rappaport, F.: *Wireless Personal Communications*. Kluwer Academic Publishers, 1993.
5. Anderson, L.: A Simulation Study of Some Dynamic Channel Assignment Algorithms in High Capacity Mobile Telecommunications System. *IEEE Trans. On Vehicular Tech.* Vol VT-22, 1973, p.210.
6. Garret, M., Fernandez, A.: Variable Bit Rate Video Bandwidth Trace Using MPEG Code. Bellcore, <ftp://ftp.bellcore.com/pub/vbr.video.trace/>, 1992.
7. Garret, M.: *Contributions Towards Real-Time Services on Packet Networks*. Ph.D. Dissertation. Columbia University. May 1993

# Quality-of-Service (QoS) in Heterogeneous Networks: CLIP, LANE, and MPOA Performance Test

Kai-Oliver Detken  
Head of wwl network Bremen

WWL Internet GmbH, Goebelstr. 46,  
D-28865 Lilienthal/Bremen, Germany  
detken@wwl.de,  
Business: <http://wwl.de>  
Private: <http://kai.nord.de>

**Abstract.** Interoperability is a most challenging issue in today's network infrastructures. Especially ATM and IP have to be ever more adaptive and integrative with each other. Additionally, the issue of QoS has not been adequately addressed in today's available heterogeneous platforms. This paper will be discussing the interoperability of different technologies such as the Internet Protocol (IP) and the Asynchronous Transfer Mode (ATM) in a heterogeneous environment, because they are considered the most important protocols in the near future. Furthermore, this paper will be discussing Classical IP (CLIP), LAN Emulation (LANE) and Multiprotocol-over-ATM (MPOA) solutions concerning the adaptation and integration of IP and ATM and their performance. As a manufacturer-independent company, the WWL Internet AG (WWL) tested different already available solutions of different manufactures and compared the results. WWL regularly evaluates new technologies for ones own better understanding and in order to build up special knowledge to support qualified customers. Following there will be, furthermore, discussed the effectiveness of the integration of IP and ATM and this will be accompanied by test results.

## 1 Introduction

In the Internet and in other networks, Quality-of-Service (QoS) is the idea that transmission rates, error rates, and other characteristics can be measured, improved, and, to some extent, guaranteed in advance. QoS is of particular concern for the continuous transmission of high-bandwidth video and multimedia information. Transmitting this kind of content dependably is difficult in public networks using ordinary best-effort protocols. QoS refers to the performance guarantees that a network can offer to its users, which in turn determines what the network can be used for. QoS is essential to the current and future markets for telecommunications.

Some examples that represent the most rudimentary attempts to provide QoS control can be found in the Internet's Resource Reservation Setup Protocol (RSVP) and in the ATM world. In the RSVP environment packets passing through a gateway host

can be expedited based on policy and reservation criteria arranged in advance. In an ATM network provision is made such that a company or user pre-selects a level of quality in terms of service. QoS can be measured and guaranteed in terms of the average delay at a gateway, the variation in delay in a group of cells, cell losses, and the transmission error rate. Latency, i.e. delay, is one of the basic parameters that determines a network's QoS. The main question to ask is then how latency and performance affects the Internet protocols? Especially the performance has been tested in this article.

## 2 Comparison of LANE and MPOA

For the adaptation or integration of IP into ATM, there are different methods existing developed by the ATM-Forum (LANE and MPOA) and IETF (CLIP and Multiprotocol Label Switching). For this paper, LANE and MPOA have been tested compared with CLIP, because they are based on each other, they are in ATM products currently available and support legacy LANs such as Ethernet and Token Ring.

LAN Emulation (LANE) is after CLIP the second solution for IP-over-ATM and can be best characterized as a service developed by the ATM Forum that will enable existing LAN applications to be run over an ATM network. In order to do so, this service has to amalgamated the characteristics and behaviors of traditional Ethernet, Token Ring and FDDI networks. Moreover, it has to support a connectionless service as current LAN stations send data without establishing a connection before the operation takes place. Therefore, LANE must support broadcast and multicast traffic such as the kind of traffic allowed over shared media LANs. It has to allow the interconnection of traditional LANs with the amalgamated LAN and at the same time to maintain the MAC address identity associated with each individual device that is attached to a LAN. Finally, it has to protect the vast installed basis of existing LAN applications and enable them to work in the same way as before over an ATM network. This may be put into practise over the OSI layer 2, whereas the LANE technology can be understood as a bridging technology.

MPOA clients established VCCs with the MPOA server components in order to forward data packets or to request information. Using these components, the client is able to establish a more direct path. MPOA supports various kinds of routable protocols (IP, IPX, AppleTalk, etc.), integrates existing internetworking protocols (RFC-1577, RFC-1483, NHRP, MARS, RSVP) and the IETF and ATM Forum solutions (LANE, P-NNI) into a virtual router environment. Yet actually, MPOA supports only IP and based on LANE, RFC-1483, and NHRP. MPOA is a service with layer 3 internetworking support for hosts attached to ELANs, ATM networks and legacy LANs. Thus, the real premise behind MPOA is to provide and deliver the functionality of a router and to take as much advantage of the underlying ATM network as possible. MPOA works as a virtual router on the OSI layer 3. [1]

## 2.1 TCP/IP-Over-ATM

IP is an important protocol used to achieve interoperability in a heterogeneous network. In contrast, the effectiveness of IP in high speed networks is not widely known. There is much doubt about the TCP performance, in particular, as the acknowledgment mechanisms, overhead size and parameter setting are considered to be obstacles. UDP is more appropriate for real-time data streams, but does not have any security mechanisms.

The TCP/IP protocol stacks were not designed for high speed performance networks in the first place. Several extensions of TCP protocols have been suggested in order to achieve higher performance over these networks and to improve the performance of connections with a high bandwidth delay. New bandwidth-intensive applications such as multimedia conferencing systems and characteristics of high speed networks have triggered research on advanced transport protocols. Therefore, the discovery of additional error sources is not surprising. Following, the reader will find factors identified by WWL that are responsible for inefficiencies of TCP protocols over ATM: [3]

- Send and receive socket buffer size
- Network: Maximum Transport Unit (MTU)
- Protocol: Maximum Segment Size (MSS)
- Transmitter: use of Nagle's algorithm
- Round Trip Time (RTT)
- Receiver: delayed acknowledgement mechanisms
- Transmitter: Silly Window Syndrome (SWS)
- Copy strategy at the socket interface
- Network congestion and lost notice

## 2.2 Scenarios for LANE and MPOA

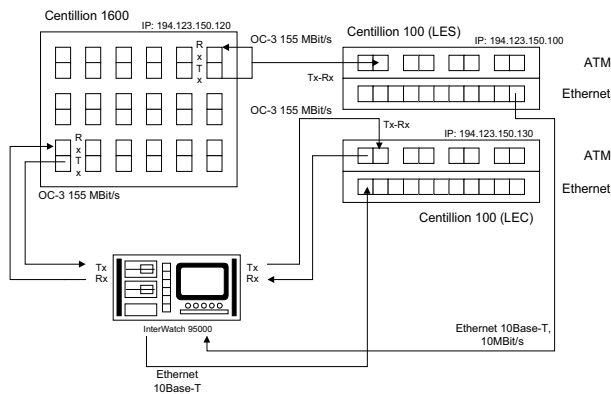
The following ATM devices were used during the tests in order to measure the performance of MPOA and LANE:

1. ATM-Switch Centillion 100 (Bay Networks) with ATM module ATMSpeed MDA MCP, software C100 SpeedView 2.2/3.0.1
2. ATM-Switch Centillion 1600 (Bay Networks) with ATM OC-3 module 155M-SMFS, software C1000 7.0(1).2
3. ATM-Switch CS3000 (Newbridge Networks) with ATM155 module (OC-3), VIVID software version 3.0 and system manager
4. VIVID Route Server (Newbridge Networks) with SC Connector and ATM MMF OC-3 155 module, Routing Protocols RIP, OSPF, NHRP and support of IP, IPX, etc.
5. VIVID Orange Ridge (Newbridge Networks) with Ethernet interface 10BASE-T Orange Ridge MMF, and ATM uplink interface.

- 6. ATM NIC from ForeRunnerLE series with throughput of 155 Mbps, bus architecture PCI 2.0/2.1, 32-Bit, 33 MHz PCI Bus and Windows95/NT 3.51/4.0, NDIS 4.0 und NDIS 5.0
- 7. ATM NIC from Olicom Rapid Fire OC-6162: 155 Mbps over MMF (SC Connector), NDIS 3.0 NIC driver

In the first step, the measurement scenarios were defined. In doing so, WWL distinguished between LANE and MPOA tests. LANE was tested without any router as there has been just one Emulated LAN (ELAN) used. This can be considered an optimal way of using LANE, because there is no performance bottleneck. The MPOA scenario was different to LANE, because Logical IP Subnets (LIS) were needed in order to use the forwarding and routing functionality of MPOA.

Nonetheless, a direct comparison regarding the effectiveness and the performance was possible, yet it has to be stressed that LANE did not use a router in the test scenario. In order to prevent a falsification of the measurement functionality, switches like Spanning Tree and dynamical routing were not tested. On the other hand, measurements with various packet sizes were carried out to make sure that performance and latency were represented correctly. Fragmentation, a process carried out by bigger packet sizes, influenced the measurements considerably. Typical packet sizes of IP are 576 Byte (Internet), 1500 Byte (Ethernet), and 9180 Byte (CLIP).



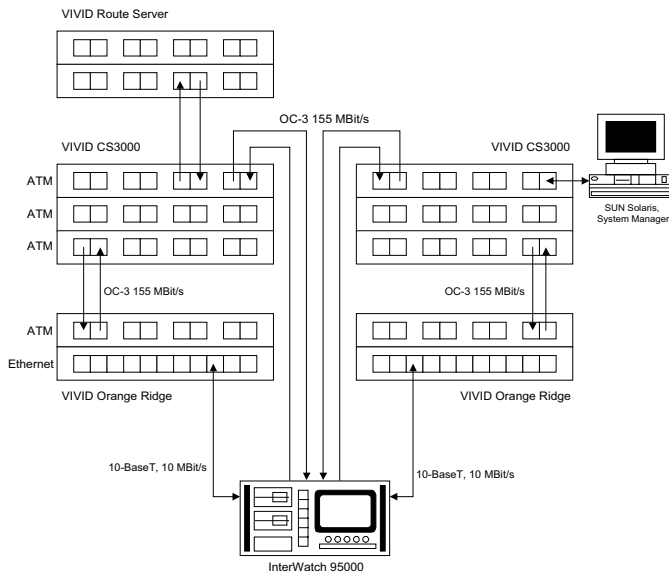
**Fig. 1.** LANE test scenario with components of Bay Networks: The BUS allows only unknown addresses to get through and these addresses were used for the measurements. The ARP-Request was not used because terminal equipment was missing at the time the tests were carried out. A problem of the benchmark tests was the sending of a ping. Finally, it was send as continuous broadcast.

LANE was tested by a pure Bay Networks scenario. The Centillion 100 has been configured as LAN Emulation Server (LES), whilst the second Centillion 100 has been configured as LAN Emulation Client (LEC). The Centillion 1600 was used as a backbone switch. In doing so, the Centillion 1600 worked in this scenario as a native ATM switch. LANE functionality is generally only to achieve by using an additional module, yet was not required for the tests.

The test device InterWatch 95000 with its ATM OC-3 interfaces was connected to the two Centillion 100 switches. As of the configuration of LES and LEC, the communication between the two has to be directly connected with the switch Centillion 1600 and the InterWatch 95000. The test device was, moreover, connected to the switches of Centillion 100 via Ethernet interfaces in order to produce data traffic and to measure the performance of the Ethernet. Using this scenario, the following tests were carried out:

- Performance Ethernet to Ethernet
- Performance ATM to ATM
- BUS performance

The measurement results differed at the very beginning, because the MAC addresses were not known right from the start of the test. A reply message was sent regularly every few minutes in order to briefly inform about the missing address solved this problem. In doing so, the performance of one link can exactly found out. Since none permanent broadcast has been sent, the results showed a better exploitation of the connection and, furthermore, only the real traffic was measured. The author would like to stress that this is a frequent mistake of benchmark tests.



**Fig. 2.** MPOA test scenario with components of Newbridge Networks: The MPOA measurements differ to the tests with the Bay Networks equipment and LANE. The Newbridge Networks equipment was used with the VIVID series. Two ATM switches CS3000 were used for the backbone. The Newbridge Orange Ridge edge devices which convert Ethernet frames into ATM cells and the other way around have been used for the MPOA functionality. Furthermore, P-NNI was configured in a way to automatically establish dynamical routing. Newbridge devices work only with LANE version 1.0. Therefore, this software was the only one available in order to establish a heterogeneous network with Newbridge components.

During the tests, MPOA operated with its specific parameters. The InterWatch 95000 was similar to the LANE configuration connected via the Ethernet ports to the edge devices VIVID Orange Ridge. Additionally, the test device was connected between the two ATM switches in order to measure the ATM performance. Summarizing, the following measurements have been carried out in performance test Ethernet-to-Ethernet and ATM-to-ATM. Using this configuration, it was possible to conduct research under the same conditions on the Route Server performance similar to the LANE BUS performance. The test procedure was repeated to gain test results which could be compared for the different methods of LANE and MPOA. [2]

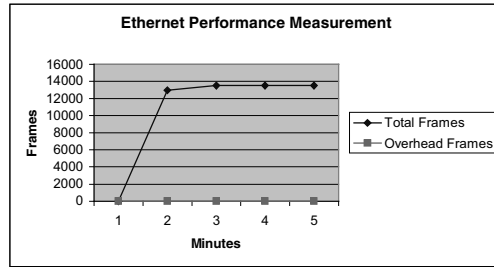
### 2.3 TCP/IP-Over-ATM Scenario

The tests were carried out over a 155 Mbps connection between two separate Pentium-II/350MHz and Pentium-II/266MHz workstations with operating systems Windows98 and Windows NT 4.0. Both clients had an ATM Network Interface Card (NIC) from Fore Systems (ForeRunnerLE155) and also the RapidFire6162 ATM 155 PCI Adapter from Olicom. The central ATM switch was Cabletron's SmartSwitch2000 with a single ATM module and four OC-3 interfaces. AAL-5 was used for all measurements to encapsulated IP packets. The sender buffer varied between 16-64 kbyte, whilst the receiver buffer was always of the same value. Classical IP (CLIP) was used for the scenario without any routing. Therefore, it was a point-to-point connection between both workstations under optimal conditions.

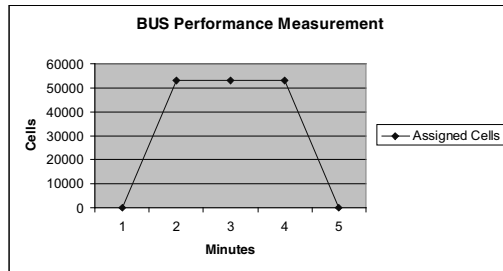
Netperf, developed by Hewlett Packard, was the software used for test-purposes on both clients. It is a benchmark program that to measure the performance of networks. Measurement opportunities for burst traffic and request/response performance are the main tasks of the program. Optional extensions of the software are available. In contrast to the full Netperf version that is only available for Unix operation systems, the used test-version was especially designed for Windows end systems.

### 2.4 LANE Measurement Results

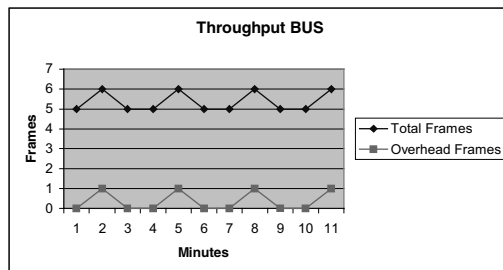
The switch Centillion 100 was started immediately afterwards in order to reactivate LANE. It comes to an interruption of the signalling packets at full load. Therefore, the switch went down, which is not a good feature in real network environment. Yet, in a real situation this will hardly happen as the BUS normally has to forward just the unknown and broadcast packets. After the establishment of the connection, the link is directly existing between LEC to LEC. Therefore, the BUS does not need to have such a performance and is able to work with short bursts. Figure 4 shows the short measurements which increase to 53.000 cells per second and go down again after a few minutes. In the next step, the exploitation was to set at 50% in order to allow a longer operation of the BUS and to measure the unknown packets. Moreover, it was considered a more real environment for a BUS operation. The measurements were stopped after a short residence time.



**Fig. 3.** In order to measure the performance of the Ethernet side, a PING was sent in both directions. In doing so, the measurements could be limited to one port. Furthermore, an exploitation of 100% was adjusted. Figure 3 clearly shows that the throughput of the Frames after a very short time climbs up to 13.500 frames/sec and remains at that level. This figures demonstrate an effective exploitation of 99,4%. Yet, the payload is included and that means you have to add some collisions (27) and the overhead.



**Fig. 4.** In a next step, the BUS performance on the ATM side was tested. The maximum BUS throughput is approx. 53.000 cells per second. This represents a data rate of 22,4 Mbps. Full duplex was chosen as the full possible data rate plus overhead had been reached. This result has got independence from the data rate and can be hold only approx. one minute.

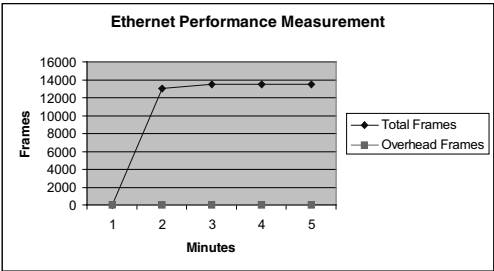


**Fig. 5.** This figure shows the unknown packet throughput that was responsible for the allocation of the addresses. The unknown packets did not load the BUS as figure 5 illustrates. The BUS sent only 5-6 frames with 0-1 frames overhead.



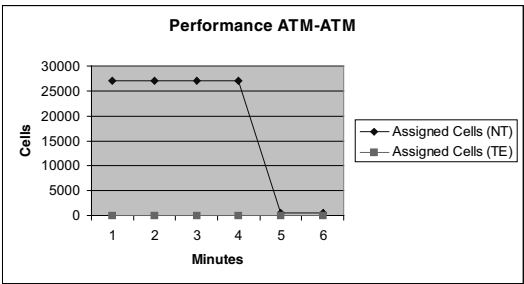
2.5 MPOA Measurement Results

The MPOA measurements could not be configured directly together with the Inter-Watch 95000, because MPOA had not been not implemented at that time. Therefore, the first tests were repeated as they were also valid for MPOA. That means, the encapsulation of the data traffic and the operation was similarly to LANE.



**Fig. 6.** Ethernet-to-Ethernet performance: The measurements showed the same result as the aforementioned with LANE. Changes of the packet sizes from 100 byte to 1000 byte and 64 kbyte did not influence the test results. The explanation to that is the 10 Mbps data rate. The exploitation was 99,4% with 1.100 collisions.

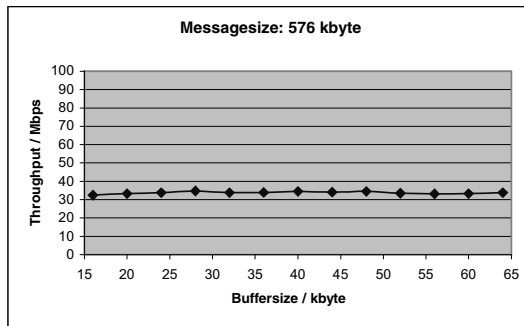
This test has been conducted with more than two subnets in order to force the router to route and to forward the packets. With MPOA the route server established a shortcut connection to directly pass the data traffic from one switch to the other. 27.000 cells as the maximum throughput was measured. This equals a data rate of 11,45 Mbps. The reason for this data rate was the unidirectional connection that was measured. Otherwise the performance was identical with LANE without router integration. In case of a network with different LIS, MPOA this may be very useful.



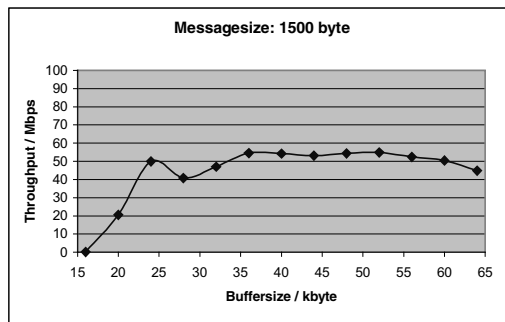
**Fig. 7.** ATM-to-ATM performance: The second measurement has been conducted with 100% exploitation. After a defined period of time, the route server also had to be restarted, similarly to the BUS at LANE.

## 2.6 TCP/IP-Over-ATM Performance

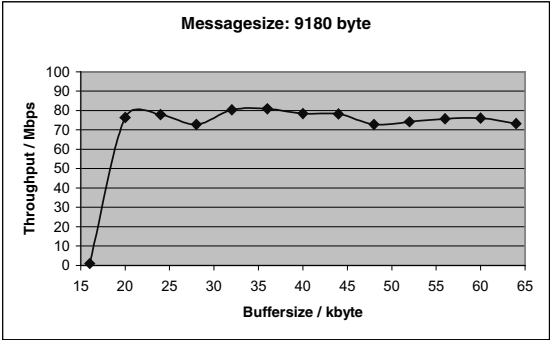
The measurement of TCP/IP-over-ATM was carried out by using various packet sizes in order to represent the effectiveness of IP-over-ATM. The throughput alone was not really interesting, because of the different bottlenecks that were already mentioned before. The fluctuations and throughput breaks were more important. Classical IP (CLIP) has been used for this measurements point-to-point, because of the highest effectiveness. The test phase had a time duration of 60 seconds for each measurement via Netperf.



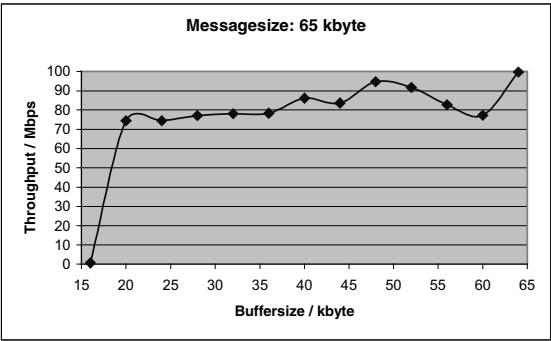
**Fig. 8.** IP-over-ATM, 576 byte: This figure shows the TCP/IP measurements with a packet size of 576 byte via CLIP. The buffer varies from 15-64 kbyte. The packet size of 576 byte represents a normal datagram in the Internet environment. Figure 8 demonstrates the effectiveness of the achieved throughput during a normal point-to-point session with minimal overhead. By the less datagram size the effectiveness went down to approx. 35 Mbps.



**Fig. 9.** IP-over-ATM, 1500 byte: Additional breaks and interruptions happened with small and big packet sizes. The MTU of 1.500 byte required a fragmentation of the packets and this has to be considered the reason for it. If the fragmentation was increased the performance went down. Figure 9 shows a differing result. In this case, the packet size was 1.500 byte which allows higher data rates then before. The results are only limited to approx. 60 Mbps. The measurements show some fluctuations, because of the different buffer sizes, which was used.



**Fig. 10.** Fluctuations happened, especially at a buffer size of 55 kbyte. The best results were achieved at 40 and 65 kbyte. A fundamentally improved performance was achieved with a packet size of 9.180 byte. The throughput increased up to 80 Mbps. Fluctuations happened, again (especially on less buffer sizes), like during the other measurements. The small fragmentation of the packets has to be considered the reason as the MTU size was 9.180 byte.



**Fig. 11.** IP-over-ATM, 65 kbyte: IP-over-ATM, 9180 byte: For the last measurement a packet size of 65 kbyte was chosen. It triggered several fluctuations as figure 11 shows. At first the throughput went down to approx. 5 Mbps. This performance has to be considered poor and was not efficient enough to be used in practice. Otherwise this results show the best data rates till 100 Mbps. Concluding, the big packets are useful for data networks with big buffer sizes and the maximum transport unit (MTU) is responsible for an effective data throughput.

The NIC RapidFire6162 from Olicom together with the LE155 from Fore Systems have been an efficient throughput and high Switched Virtual Circuit (SVC) performance. The LE155 has been also an active processor for direct traffic support without influence the CPU of the workstation. Both NICs need 128 Mbyte for a efficient operating. Additionally, both cards support in MPOA mode different service classes as Constant Bit Rate (CBR), Variable Bit Rate (VBR), and Unspecified Bit Rate (UBR). Available Bit Rate (ABR) was only available at the RapidFire6162. But, for the support of ABR you need this service inside the ATM switch, which was not integrated in every component yet.

### 3 Conclusions

There are basically three fundamental elements needed for a successful QoS implementation. At first, QoS within a single network element (for example, queuing, scheduling, and traffic shaping tools). Second, QoS signalling techniques for coordinating QoS from end-to-end between network elements. And third, QoS policy, management, and accounting functions to control and administer end-to-end traffic across a network. Three basic levels of end-to-end QoS can be provided today:

- a) *Best-effort service* – also known as lack of QoS, best-effort service is basic connectivity with no guarantees.
- b) *Differentiated service (statistical QoS)* – some traffic is treated better than the rest. This is a statistical preference, not a hard and fast guarantee.
- c) *Guaranteed service (guaranteed QoS)* – an absolute reservation of network resources for specific traffic.

The type of service that is needed depends on the application or problem the user is trying to overcome. There are applications where a best-effort service may be sufficient. There are also problems such as real-time control over a network where the guaranteed service is absolutely necessary. It may also happen that a best-effort customer will migrate to differentiated service and then to guaranteed service. In many applications there may exist an upgrade path from the technology needed to provide differentiated services to that needed to provide guaranteed services. Network performance is influence e.g. by traffic offered and thus network performance should be taken as a random variable and has been analysed from this point of view in this article.

LANE is a very stable standard and simply to configure. The components of Bay Networks showed a reliable and efficient performance. If there was no router bottleneck, LANE had the same performance as MPOA, because of the direct connections between the two LECs provided the BUS found the correct addresses. If there was a router in the LAN, MPOA was necessary. As LANE version 1.0 supports only Unspecified Bit Rates (UBR) no Quality-of-Service parameter could be used. MPOA will solve this problem, because the version 1.1 will support service classes.

The missing standard conformity in software 3.0 of the VIVID components limited the possibility of using all defined functionality of MPOA. Only the Next-Hop Resolution Protocol (NHRP) and LANEv1.0 were supported. Additionally the edge device Orange Ridge had only one ATM uplink that did not allow redundant connections. The management software was not easy to install and only available for Solaris 2.5.0. Newer Solaris-versions were not supported. Furthermore, the series of VIVID used a propriety protocol for the management with the name Control Packet Switching System (CPSS). Other components and devices by other manufactures were not supported and could not even be recognized by the management software. Summarizing, the VIVID components were highly reliable and showed a good performance.

TCP has been extended and further developed for better and more efficient mechanisms in high-speed networks. Yet in practice, TCP quite often does not have the op-

timal throughput. Several parameters, devices and mechanisms were responsible for this effect. This caused fluctuations and delays during the transmission of data traffic. Mainly TCP's own mechanisms were responsible for the small throughput such as acknowledgements, small window sizes and sequence number overloads. Additionally, the performance of the ATM boards and the end stations were important. The adapter boards from Fore Systems and Olicom showed only a maximum throughput of approx. 100 Mbps, if you use CLIP. Earlier measurements conducted in some European projects (e.g. European Information Exchange Services – EIES) got values of 117 Mbps at LANE and 134 Mbps at CLIP with high buffers and two Sun Solaris workstations. Therefore, there are three main issues regarding latency in Internet protocols, which are able to improve the performance:

1. The default “window size” in many TCP/IP protocol implementations acts as a bottleneck on communications over high-latency links. On many implementations, the standard window prevents sending enough data to fill a high-latency connection.
2. TCP includes two essential congestion control mechanisms called “slow start” and “congestion avoidance”. These mean that all Internet connections (such as viewing web pages and sending e-mail) start out at lower bandwidth and then throttle up to higher speed if no congestion is encountered. The problem is that each cycle of speed increase requires a full round-trip communication between sender and receiver, and dozens of such round-trips can be necessary to reach the full potential of a link.
3. There are research efforts to look at increasing the performance of TCP by tricking the connection on the other side into believing it is communicating over a low-latency link. Unfortunately, these schemes fundamentally alter the semantics of TCP communications, introducing the possibility of data corruption. Moreover, they are incompatible with the IP security protocols (IPsec), which promise to bring an unprecedented and badly needed degree of security to the Internet

In order to implement IP-over-ATM effectively, it is necessary to assign enough time to the configuration of the participating devices and software. ATM is a new technology which considerably extends the usable bandwidth with regard to QoS parameters. Nowadays, workstations are designed for work in traditional networks such as Ethernet or Token Ring. These deadlocks must be compensated in order to use efficiently IP-over-ATM.

## References

1. Detken, K.-O.: Interworking in heterogeneous environment: Multiprotocol over ATM (MPOA); European conference Interworking98; Ottawa 1998
2. Detken, K.-O.: ATM Handbook; publishing house Hüthig; Heidelberg 1999
3. Detken, K.-O.: ATM in TCP/IP environment: adaptations and effectiveness; European conference ATM Traffic Symposium; Mykonos 1997

# Experiments with Dynamic Multiplexing and UPC Renegotiation for Video over ATM

Maria Teresa Andrade and Artur Pimenta Alves

INESC Porto, Praça da República, 93 R/C, Porto, Portugal

FEUP, DEEC, Rua dos Bragas, Porto, Portugal

telephone: +351 22 2094238 fax number: +351 22 2084172

{mandrade, palves}@inescn.pt

<http://telecom.inescn.pt>

**Abstract.** In this paper we present an experimental approach for QoS-aware and bandwidth-efficient transmission of multimedia sources over ATM networks. VBR video sources are statistically multiplexed at the application level and the UPC parameters of a single connection for the aggregate traffic are dynamically renegotiated with the network during the session lifetime. A statistical multiplexer computes the required bandwidth for the aggregate traffic and dynamically assigns bandwidth according to the performances requested and network resources availability. A dynamic UPC manager using feedback information sent by the network, may initiate a renegotiation of traffic parameters upon receiving a request from the statistical multiplexer. Performing the connection admission control procedures for the aggregate traffic, allows significant reductions in the value of the total peak rate when compared to the sum of peak rates of individual connections. It also reduces the burstiness of the flow submitted to the network, thus increasing the lifetime of each set of UPC parameters.

*Keywords:* Quality of Service, ATM, UPC, statistical multiplexing, renegotiation

*Abbreviations:* **AAL**, ATM Adaptation Layer; **ABR**, Available Bit Rate; **ACTS**, Advanced Communications, Technologies & Services; **ATM**, Asynchronous Transfer Mode; **CAC**, Connection Admission Control; **CBR**, Constant Bit Rate; **CDV**, Cell Delay Variation; **CDVT**, Cell Delay Variation Tolerance; **CTD**, Cell Transfer Delay; **EFCTI**, Explicit Forward Congestion Indication; **GOP**, Group Of Pictures; **MBS**, Maximum Burst Size; **MPEG**, Motion Picture Expert Group; **NIC**, Network Interface Card; **PCR**, Peak Cell Rate; **MCR**, Minimum Cell Rate; **QoS**, Quality of Service; **RM**, Resource and Management; **SCR**, Sustainable Cell Rate; **UNI**, User Network Interface; **UPC**, Usage Parameter Control; **VBR**, Variable Bit Rate; **VC**, Virtual Circuit; **VoD**, Video on Demand.

## 1 Introduction

ATM and MPEG2 are key technologies for the deployment of high-quality networked video services in the present multimedia communications scenario. The use of MPEG2 allows economies in transmission bandwidth while providing a high-quality service. The video compression algorithm reduces the number of bits necessary to represent the video sequences by exploiting spatial and temporal redundancy. However, because redundancy of sequences is variable, constant quality is only obtained if this number of bits is not restricted to a given value.

Transmission of constant quality compressed video streams and therefore variable bit rate streams, is possible in broadband networks using the Asynchronous Transfer Mode. Traditional transmission channels offer only constant bit rate connections which either leads to an inefficient bandwidth utilisation or to varying picture quality. For the compression of video sequences, high-activity scenes will require more bits to achieve the same quality level as low-activity scenes. Also, within the same scene, some pictures will present more motion than others. Although more frequent, these fast-rate *intra-scene variations* present smaller fluctuations compared to the less frequent *inter-scene variations*. These time-varying requirements allied to the objective of efficient network usage calls upon the use of VBR channels. However, the use of VBR connections is not by itself an assurance that the requirements of the bit stream will be satisfied for the whole duration of the call together with an efficient usage of network bandwidth. Traffic parameters chosen at connection setup to closely match the initial characteristics of the video, may no longer be suitable after a scene change.

A dynamic UPC with adjustable renegotiable parameters is able to contribute to a more efficient use of bandwidth while meeting the requirements imposed by the low-frequency bit rate variations. Because the renegotiation process imposes a rather significant overhead to the network and because there is a latency in the response from the network, the renegotiation frequency should be kept as small as possible. It should only be triggered if the currently available bandwidth is not sufficient to guarantee the minimum target quality for all sources. Also, It should only be allowed at specified regular intervals. On the other hand, it should be controlled through the use of network feedback information. In this paper we propose to implement this control by using the explicit rate and congestion indication mechanisms specified in the ATM ABR service class.

To deal with the intra-scene variations we use the statistical multiplexer which aggregates a collection of VBR video sources and dynamically assigns bandwidth, in order to satisfy as fair as possible the level of quality indicated by each source. The objective is to obtain increased overall statistical gain while improving the quality of service for an aggregation of video sources.

## 2 Motivation and Applicability

Statistical multiplexing algorithms have been extensively studied and used to allow savings in overall transmission bandwidth in packet networks. Literature

in this field is quite representative of the work already developed [2] ... [7]. More recently, researchers have started to study the potential gain, both in effective bandwidth and quality improvement, in the transmission of video signals over packet networks using adjustable UPC and traffic parameters renegotiation [8] ... [13]. However the effects of using the two techniques combined have not yet been sufficiently explored. In [14], [15] and [16], the authors present approaches and results which demonstrate the benefits of multiplexing a set of video sources together with renegotiation of the traffic parameters of a single network connection for the aggregate stream. In our work we try to jointly explore the benefits of performing statistical multiplexing at the user level and dynamic adjustable UPC based on the following premises:

1. independent video sources are highly uncorrelated;
2. constant quality of compressed video is only possible with VBR;
3. constant quality compressed video presents two types of rate variability: low-amplitude/high-frequency *intra-scene* variations and high-amplitude/low-frequency *inter-scene* variations (respectively short-term and long-term in [2]);
4. renegotiation of traffic parameters provides a way of matching more closely the application requirements, but it imposes significant overhead to the network and consequent latency in the response of the network to those requests;
5. the same user can combine in a single ATM connection one or more video channels and share at his will the overall requested bandwidth while having to control one single network access unit;
6. a reduction in the number of renegotiation requests reduces the blocking probability of the network (or improves the network performance).

Of great importance is the existence of applications which may actually benefit from the use of the proposed approaches.

We believe there are areas of potential application for the combined use of statistical multiplexing and dynamic renegotiation of traffic parameters, such as:

- in TV broadcasting, a supplier of TV programmes may use an ATM network as the backbone between a National Studio Centre and National terrestrial transmitters. It may hire a single channel and send through it 2 or more

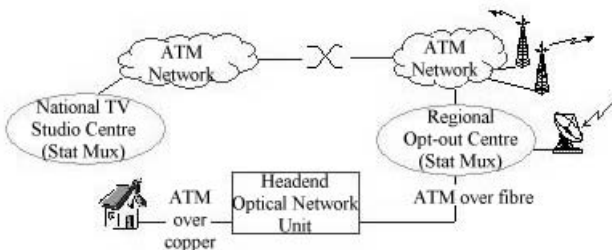


Fig. 1. Example of an application scenario



TV programmes. The bandwidth it negotiates for with the network is dynamically shared among those programmes according to the user allocation bandwidth algorithm. The TV supplier has thus the freedom to choose and change at his own will the combination and number of simultaneous programmes and the distribution of bandwidth. Some work in this field was already developed in the ACTS project ATLANTIC [3];

- in TV studio production: journalists, editing producers, archivists siting at workstations interconnected by a local ATM network, access a/v material stored in networked distributed servers. Each user requests a single VBR connection and has a single connection manager running locally in his machine. That single VBR channel is then shared dynamically by different video and audio sources. For example, a journalist when preparing the skeleton of his article will most certainly have to access a great number of different video and audio sources in a non-linear, non-sequential way. His local connection manager, comprising a statistical multiplexer and dynamic UPC manager will ensure that the required bandwidth to perform the desired operations will be available (if granted by the network) and shared in the best possible way. TV studio operations similar to these ones have already been experimentally implemented in the ACTS project ATLANTIC and in the framework of a Portuguese funded research project VIDION [4];
- in infotainment applications where users combine and interact with distinct video sources and other types of media, in a completely unstructured way;
- a supplier of VoD services, using an ATM backbone to interconnect remote islands of users to the central server where films are stored and accessed.

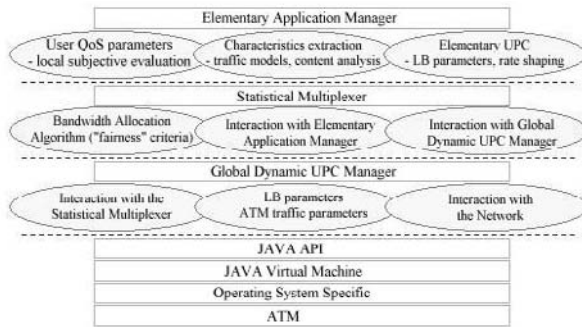
### 3 Description of the Experiments

#### 3.1 Overview

The generic system concept is depicted in figure 2. All modules in our system implement *closed-loop control algorithms*, in the sense that they all react to information they receive from other modules.

The *statistical multiplexer* implements a dual closed-loop bandwidth allocation algorithm which reacts to requests from the sources and to feedback information from the global dynamic UPC. Likewise, the UPC managers use a dual closed-loop parameters adjustment algorithm. The elementary UPC interacts with the real-time VBR video encoder and with the statistical multiplexer. The global UPC manager interacts with the network, receiving feedback information regarding network resources availability and with the statistical multiplexer.

Real-time VBR video encoders also use a closed-loop rate control mechanism to increase or decrease the rate of generation of bits. The rate adaptation is performed by modifying accordingly the values of the quantisation step sizes. Each real-time video source is VBR encoded and can be either an *on-line* (encoded on-line as it is being transmitted) or *off-line* (pre-encoded and stored material) video source. Our experiments have been restricted to the *off-line* case.



**Fig. 2.** Conceptual architecture of the system

Applications connect themselves to the system via their application manager, by supplying quality of service parameters from the user perspective: target picture quality (TPQ), minimum picture quality (MPQ) and service type (ST). With these parameters and a sample analysis of the bitstream to extract associated peak and mean bit rates, the application manager conveniently selects the parameters of the elementary UPC and passes the bit rate information and the user MPQ, TPQ and ST parameters to the statistical multiplexer. This entity estimates necessary values of peak and mean bit rates to allow the transmission of the aggregate traffic from all active sources. It passes this information to the user-to-network interface module, which then calculates the appropriate parameters of a global UPC capable of monitoring the aggregate traffic. It also specifies the ATM QoS and traffic parameters to request an ATM channel adequate to support the transmission of the multiplexed stream.

### 3.2 Elementary Application Managers

The application manager of *off-line* video sources extracts from the stored data, information concerning bit rates and service type. This information is directly used to select the parameters that better match the characteristics of the source. For *on-line* sources, there can be two alternative methods to obtain the same information: manually, through the intervention of a human, who supplies the desired user QoS parameters and bit rate information; automatically, with the application manager analysing a sample number of frames of the video source and using a parameterised set of values for the objective picture quality.

Once the desired user QoS and bit rates are specified, the parameters for a flow rate monitoring mechanism are calculated by the application manager. This monitoring is performed by an elementary dual Leaky Bucket mechanism the way as proposed by the ATM FORUM [1]. It comprises two LB, each one implemented through a finite-capacity buffer, the token buffer, with defined sizes and drain rates, B and R, according to the desired peak and sustainable rates, length of bursts and spacing between bursts. One of the LBs monitors the peak

rate with parameters  $R_p$  and  $B_p$  and, the other monitors the sustainable rate with parameters  $R_s$  and  $B_s$ . The values of the LBs parameters are related with the traffic parameters in the following way:

$$\begin{aligned} R_p &= PCR & B_p &= 1 \\ R_s &= SCR & B_s &= R_p * T - b \\ MBS &= R_p/T & MBS &= 1 + \text{int} \left[ \frac{\min(T_1 - T_s, t_s)}{T_s - T} \right] \end{aligned}$$

$b$ : number of tokens drained from the token buffer  $B_s$  in a frame period  $T$ ;

$T_1$ : minimum spacing that bursts of size equal to MBS must observe to be in conformance;

$t_s$ : burst tolerance.

The dual LB constraints the source such that its maximum and average bit rates do not exceed the negotiated PCR and SCR and that the length of bursts (number of consecutive cells transmitted with the minimum inter-cell spacing  $T_p = 1/R_p$ ) does not exceed  $MBS = (R_p \times B_s)/(R_p - R_s)$ . The token buffers, initially set to their capacity  $B_p$  or  $B_s$ , are continuously drained at a constant rate  $R_p$  or  $R_s$  and filled at the rate of submission of cells to the network. A cell may only be submitted to the network if it is able to add a token into the token buffer. If the buffer is full, the cell is not in conformance with the UPC and therefore should not be transmitted. The traffic parameters PCR, SCR and MBS are continuously evaluated for each GOP and their values set to:

*PCR = bit rate of the largest picture in GOP*

*SCR =  $(1 - k) \times R_{avg} + k \times PCR$  with  $0 < k < 1$*

*$R_{avg}$  = weighted sum of bit rates of all but the largest frame in the GOP*

If significant changes are detected, the application manager requests a modification in the allocation of bandwidth to the statistical multiplexer.

We have performed experiments for stored video using accompanying metadata files with the following information: average and peak bit rates for each scene, initial and ending times of each scene (or the period during which the previous values are valid), frame period GOP structure and target picture quality. If this metadata is not available, the application manager must calculate them throughout the session lifetime or completely at connection set up.

### 3.3 Statistical Multiplexer

Statistical multiplexing, the ability of dynamically allocating bandwidth among sources according to picture activity and target quality, is based on the theory that independent video sources are likely to exhibit the same amount of activity simultaneously and therefore, the need for the same amount of bits.

Statistical multiplexing aims at providing better channel utilisation and better picture quality across a number of video sources. The way it proposes to

achieve this goal is to dynamically distribute the overall channel bandwidth, assigning variable amounts of bit rates to each source according to their present activity (complexity and motion). Statistical multiplexing has been traditionally applied to the use of a fixed-bandwidth channel by a group of VBR sources [17] and [18] and inside ATM networks to efficiently support a number of different connections (VCs) simultaneously [19] and [20].

To devise a methodology for the estimation of values for aggregate statistical bandwidth and UPC+traffic parameters, we have used traces of MPEG encoded video sequences. Compression video algorithms, such as those described in MPEG standards [21,22], operate on a frame basis and use 3 different picture types: I (Intra), P (Predicted) and B (Bi-directional). Encoded pictures, belonging to one of the 3 possible types, succeed repeatedly in a structured way, the GOP structure. The amount of bits generated at the output of the encoder reflects this structure. Typically, within this GOP structure, I frames require more bits while B frames are those which require less bits.

The traces we have used are publicly available by Oliver Rose from the University of Wuerzburg ([ftp-info3.informatik.uni-wuerzburg.de/pub/MPEG](http://ftp-info3.informatik.uni-wuerzburg.de/pub/MPEG)). Each encoded sequence presents the following characteristics:

40 000 frames at 25 Hz ( $T = 40ms$ ) ;  $384 \times 288$  spatial resolution; GOP structure with  $N=12$ ,  $M=3$ ;  
VBR encoded, fixed quantisation step sizes of 10, 14 and 18 respectively for I, P and B frames.

Different transmission scenarios have been considered: *sources transmitted separately both with a fixed set of UPC parameters for the whole stream and using renegotiation; the same, but using a GOP buffer to smooth the elementary bit rate prior to submit the stream to the network; association of different number of sources, considering both GOP-aligned and not-GOP-aligned and using a fixed set of UPC parameters; the same but using renegotiation of UPC parameters.*

Table 1 presents the elementary sources behaviour in terms of mean and peak bit rates and degree of variability in those bit rates. Statistics performed at the GOP level assume that each application manager has a buffer of sufficient capacity to be able to send all but the largest picture in the GOP at the SCR. The picture with the highest bit rate observed in the GOP is sent at that maximum rate, the PCR. In table 1, the *average bit rate in GOP* is calculated using the bit rates of all but the largest picture in the GOP, while the *max bit rate in GOP* is set to the bit rate of the largest picture.

Table 2 characterises the behaviour of aggregate traffics in the same terms as for individual sources. It presents measures for combinations of different number of sources and assuming alignment or not at the GOP level. It also compares the bit rates required by those multiplexed bit streams with the bit rates required by the sum of corresponding individual sources, thus allowing the identification of statistical gain. It can be seen that using the same amount of rate variability as the threshold for initiating a renegotiation of the traffic parameters, the number of requests significantly decreases when 4 or more sources are multiplexed before accessing the network. When sources are not aligned at the GOP level, which we

**Table 1.** *Statistics of elementary sources*

<i>sources</i>	<i>bit rate values (Mbps)</i>				<i>number of variations in bit rate</i>					
	<i>frame basis</i>		<i>GOP basis</i>		<i>frame basis</i>		<i>GOP basis</i>			
	<i>avg</i>	<i>max</i>	<i>avg</i>	<i>avg of max</i>	<i>25%</i>	<i>50%</i>	<i>in avg rate</i>		<i>in max rate</i>	
<i>asterix</i>	0.559	3.68	0.448	1.78	27042	18954	1049	380	338	104
<i>dino</i>	0.327	2.00	0.232	1.38	25886	18496	770	244	179	42
<i>simpsons</i>	0.588	3.52	0.487	1.70	27185	21238	683	219	275	69
<i>lambs</i>	0.183	3.53	0.113	0.95	26058	18360	1050	411	179	47
<i>movie</i>	0.357	4.31	0.259	1.44	28407	26970	1460	729	612	253
<i>bond</i>	0.608	6.12	0.474	2.08	26980	26618	439	126	212	46
<i>starwars</i>	0.233	3.12	0.154	1.10	26591	19930	1331	594	219	50
<i>termin</i>	0.273	1.99	0.935	1.99	28663	23322	1321	466	463	92
<i>mTV-2</i>	0.615	5.73	0.512	1.75	28195	26429	1213	505	653	204
<i>mTV-1</i>	0.495	6.29	0.400	1.54	27207	20124	1198	504	596	213
<i>race</i>	0.769	5.06	0.659	1.98	25335	16256	511	119	272	61
<i>ATP-tour</i>	0.547	4.77	0.425	1.89	26827	19433	659	217	263	99
<i>soccer</i>	0.678	4.68	0.560	1.98	27269	26752	427	126	273	86
<i>sbowl</i>	0.588	3.52	0.487	1.70	27185	21238	683	219	275	69

**Table 2.** *Statistics of combined sources on a GOP period basis*

Combination of sources			Average rate			Maximum bit rate			
			bit rate value	variations		avg of peak rate	max of peak rate	variations	
				25%	50%			25%	50%
all	Sum	5.42	12794	4858	22.40	57.44	5001	1434	
	mux	A	5.24	22	1	22.87	30.97	0	0
		NA	6.56	8	1	8.35	14.35	29	3
12	Sum	5.06	10142	3799	20.17	54.03	4127	1293	
	mux	A	5.06	31	1	20.17	27.99	3	1
		NA	5.95	20	1	7.641	13.66	108	7
8	Sum	2.82	7838	3116	12.81	37.25	3032	1008	
	mux	A	2.82	247	17	12.81	19.252	23	1
		NA	3.54	86	3	5.345	11.104	144	29
6	Sum	1.95	1846	2500	9.522	25.234	1846	614	
	mux	A	1.95	247	24	9.522	14.747	40	1
		NA	2.469	92	7	3.813	8.307	218	32
4	Sum	1.587	3938	1570	7.130	1756	1392	1518	
	mux	A	1.587	279	28	7.130	11.287	62	3
		NA	1.927	145	15	3.295	6.934	165	20
2	Sum	0.537	1819	624	3.863	9.8	517	146	
	mux	A	0.537	538	98	3.863	8.386	154	15
		NA	1.038	427	60	2.579	6.549	173	21

understand as being the more frequent situation, larger gains in bandwidth are obtained. However, variability in peak rate is more pronounced when sources are aligned at the GOP level. Rate variability is the % of change relative to the values in use. It can be noted that the inclusion of extra sources in the multiplex contributes on average with a reduction of 5% in the number of renegotiation requests. The values of rate variability (in %) used to trigger the renegotiation are 25 and 50 %.

The estimation of bandwidth required for the multiplexed stream uses a simple algorithm that assumes sources are not aligned at the GOP level and that a reduction from 10% to 5% of the sum of bit rates is achieved for each new source joining the multiplex (see table 3). These values were obtained in the measures performed with the MPEG traces.

The statistical multiplexer continuously performs the bit rate calculations. If significant changes occur and on a n-GOP basis (i. e., every  $n * 12$  picture periods), it requests the dynamic UPC manager to alter the global UPC parameters. In that situation, the UPC manager may initiate with the network, negotiations for a new traffic contract (see subsection 3.4). After receiving a confirmation of the acceptance or rejection of the new traffic parameters, the UPC manager notifies the statistical multiplexer and adjusts its own policing parameters. If the new values were accepted, the statistical multiplexer distributes the bandwidth as requested. Otherwise it re-distributes the available bandwidth based on a fairness criteria, regarding values of bit rate requested and target quality level. If the bandwidth granted is not enough to completely satisfy all requests, the allocation algorithm starts by assuring that the minimum picture quality is maintained in all sources and then distributes the remaining bit rate according to a ratio between bandwidth requested and target picture quality.

**Fairness allocation bandwidth algorithm.** The statistical multiplexer uses a *fairness allocation bandwidth algorithm* based on the following definitions, measures and operations:

$$LB_{requested} = \sum max_i(1) \times (1 - k)$$

$LB_{network}$  → bandwidth actually granted by the network

$max_i(1)$  → maximum bit rate requested by source i to achieve the target picture quality level QoS(1)

$max_i(2)$  → maximum bit rate requested by source i to achieve the minimum picture quality level QoS(2)

$k$  → defines the multiplexing gain and depends on the the number of sources in the multiplex. Values for this parameter, obtained experimentally using the mentioned traces are given in table 3

*Pseudo code*

```
if ( $LB_{network} \geq LB_{requested}$ ) /* assign bandwidth as requested
for( $i = 1; i < numberOfSources; i++$ )
```

```

     $max_i assigned = max_i(1);$ 
if ( $LB_{network} < LB_{requested}$ ) /* applies reduction factor
{
     $reduction_1 = (LB_{network} - LB_{requested}) / LB_{requested}$  ;
    for( $i = 1; i < numberOfSources; i^{++}$ )
    {
        if( $(1 - reduction_1) \times max_i(1) < max_i(2)$ ) /* assign to guarantee QoS(2)
             $max_i assigned = max_i(2)$  ;
        else  $max_i assigned = (1 - reduction_1) \times max_i(1)$  ;
    }
     $\sum max_i assigned = 0$  ;
    for( $i = 1; i < numberOfSources; i^{++}$ )
         $\sum max_i assigned += max_i assigned$  ;
    while( $(1 - k) \times \sum max_i assigned > LB_{network}$ )
    {
        if( $(1 - k) \times \sum max_i assigned > 1.5 \times LB_{network}$ )
            ! requests renegotiation to UPC !
         $reduction_2 = (\sum max_i assigned - LB_{network}) / LB_{network}$  ;
        for( $i = 1; i < numberOfSources; i^{++}$ )
        {
            if( $max_i assigned > max_i(2)$ )
            {
                if( $(1 - reduction_2) \times max_i assigned < max_i(2)$ )
                     $max_i assigned = max_i(2)$  ; /* assign to guarantee QoS(2)
                else  $max_i assigned = (1 - reduction_2) \times max_i assigned$  ;
            }
        }
         $\sum max_i assigned = 0$  ;
        for( $i = 1; i < NumberOfSources; i^{++}$ )
             $\sum max_i assigned += max_i assigned$  ;
    }
}

```

The set of calculations and operations are indicated for the distribution of the peak bit rate of the aggregate connection. The case we are considering, assumes that each elementary source, if transmitted isolated, would pass through a GOP-buffer before accessing the network and would have a peak rate set to the bit rate of the largest frame in the GOP.

### 3.4 Dynamic Renegotiable UPC

In ATM networks, the establishment of a new connection must be preceded by the negotiation of a service contract between the new application and the network. The contract must identify the ATM service attributes requested from the network, through specification of traffic parameters (PCR, CDVT, SCR and MBS) and QoS parameters (peak-to-peak CDV, max CTD and CLR).

**Table 3.** *Experimental values for multiplexing gain*

<i>Number of sources</i>	<i>k</i>
2	0.30
4	0.50
6	0.60
8	0.70
12	0.75

Once the connection is accepted, the network will provide the requested QoS for those cells that conform to the negotiated traffic values. If the user violates this traffic contract, the network no longer has the responsibility of maintaining the agreed QoS. It is therefore the interest of the user, to ensure that all cells submitted to the network do not exceed the negotiated values.

Conformance of the flow submitted to the network is monitored by an UPC mechanism implemented through a dual Leaky Bucket algorithm (see section 3.2). The global UPC manager sets its parameters according to the request sent by the statistical multiplexer. At connection set up, it negotiates with the network a traffic contract specifying the ATM traffic parameters PCR, SCR and MBS for the aggregate stream and selecting the specified QoS class 2. This specified QoS class supports a QoS that meets the performance requirements of the ATM service class suitable for real-time VBR video and audio applications [1]. Formulas for calculating the traffic parameters have already been presented in section 3.2.

For each GOP period, consisting of 12 frame periods, the PCR is used during a frame period and the SCR during the rest of the GOP. This way there is a constant spacing of 11 frame periods between bursts. The values of the UPC are therefore calculated in order to match the requirements of the multiplexed stream as indicated by the statistical multiplexer. When a request for altering those parameters is received, the dynamic UPC before actually updating them, interrogates the network about the availability of resources.

Our approach to implement the interaction between the dynamic UPC and the ATM network is to use RM cells in a similar way as specified for the ABR service category [1]. The network access interface of the multiplexed source sends to the network RM cells with the indication of target bit rates. There are no values specified for the transmission frequency of those cells but typically, reported implementations send one RM cell for each group of 32 or 48 service cells. In response, the network sends feedback information in backward RM cells. This feedback consists of information about network bandwidth availability and state of congestion and can be explicitly stated in the fields *Explicit Rate*, *Congestion Indication* and *No Increase* of backward RM cells. The ER field of forward RM cells is used by the source to specify the target bit rate (the PCR). This value may be reduced by any network element while the cell is travelling from source to destination and again back to the source. Its final value when reaching the source dictates the highest rate that may be used by the source. RM cells can be



of two types: *in-rate* or *out-of-rate* cells. The first type is included in the bit rate allocated to the connection, while the later is not. *Out-of-rate* forward RM cells are not taken into account to compute the source rate, but they may not be sent at a rate greater than 10 cells/s (the TCR, Tagged Cell Rate). As we propose to restrict the renegotiation frequency to once every  $n \times$  GOP this rate of RM cells may prove to be acceptable. It is not possible however to report results in this field, namely of performance and feasibility of the proposed method and network blocking probability to renegotiation requests, because experiments have not yet been performed. It is however the intention of the authors to proceed the work in this direction.

## 4 Results and Further Work

We have evaluated our experimental system using traces of pre-encoded MPEG sequences. We have thus only implemented and off-line experiment of our proposed methodology, where the entire bitstreams to be submitted to the network were known in advance. Although having an argueably restricted field of application, this experience has revealed itself very important:

- it provided a way of evaluating the effectiveness of the proposed system:
- it provided valuable information to infer the methodology to use with on-line cases, where video sequences are being real-time encoded as they are being submitted to the ATM network.

We are developing an API in Java to provide applications the kind of functionality we have described for the *elementary application managers* combined with the dynamic renegotiation of UPC parameters. Presently, this API provides only basic functionality for native ATM applications to request an ATM connection and specify ATM QoS and traffic parameters.

In our premises at INESC, we have installed a 155 Mbit/s private ATM network with two FORE switches and a number of end stations equipped with 25 and 155 Mbit/s Fore ATM NICs. Using this infrastructure, we have set up a platform to test several TV studio applications developed within the framework of already mentioned research projects [3] and [4]. The platform allows the experimentation of the application scenario *TV studio pre and post-production*, described in section 2. This scenario involves both pre-encoded material, thus the *off-line* case, as well as *on-line* real-time encoded sources. Our work will evolve towards the experimentation of the *on-line case*. For that matter we will use, once finalised, the full-featured API for native-ATM QoS-aware applications within the TV studio applications scenario. We will also produce experiments to further investigate the feasibility of using VBR/CBR service class combined with RM cells.

The results we have obtained so far can be extracted from tables 1 and 2. They show us that it is possible to obtain statistical gain ranging from nearly 75% to 15% for the same picture quality objectives, depending on the number of sources being multiplexed. Also, that the frequency of renegotiation, initiated by

the occurrence of the same amount of bit rate variation, is significantly smaller when sources access the network in a multiplex rather than individually.

## References

1. ATM Forum Technical Committee: Traffic Management Spec Version 4.0 (1996)
2. N. Ohta, "Packet Video: Modeling and Signal Proc.", Artech House, Boston (1994)
3. AC078, ATLANTIC, Working documents, <http://www.bbc.co.uk/atlantic>
4. VIDION project, <http://newton.inescn.pt/projects/vidion>
5. O. Rose, "Statistical properties of MPEG video traffic and their impact on traffic modeling in ATM systems", Univ of Wurzburg, Inst of Computer Science (1995)
6. B. Maglaris *et al.*, "Performance Models of Statistical Multiplexing in Packet Video Communications", IEEE JSAC, Vol. 36 (1988)
7. D. Heyman *et al*, "Statistical analysis and simulation study of video teleconferencing traffic in ATM networks", IEEE Trans. Circ. & Sys. for Video, Vol. 2 (1992)
8. D. Reininger, Y. Senda and H. Harasaki, "VBR MPEG-2 Encoder for ATM networks with UPC renegotiation", NEC USA, C&C Research Lab (1996)
9. D. Reininger, D. Raychaudhuri and J. Hui, "Bandwidth Renegotiation for VBR Video Over ATM Networks", IEEE JSAC, Vol. 14, No. 6 (August 1996)
10. D. Reininger, D. Raychaudhuri and R. Siracuse, "Video Transport in ATM networks: a systems view", NEC USA, C&C Research Lab (1995)
11. B. Mark and G. Ramamurthy, "Joint source-channel control for real-time VBR over ATM via dynamic UPC renegotiation", NEC USA, C&C Research Lab (1996)
12. B. Mark and G. Ramamurthy, "Real-time estimation and dynamic renegotiation of UPC parameters for arbitrary traffic sources in ATM networks", IEEE/ACM Transactions on Networking, Vol. 6, No. 6 (December 1998).
13. S. Giordano and J. Le Boudec, "The renegotiable variable bit rate service: characterisation and prototyping", EPFL ICA, Lausanne, Switzerland (1998)
14. E. Fulp and D. Reeves, "Dynamic bandwidth allocation techniques", North Carolina State Univ, Depts of ECE and CSC (1997)
15. Y. Nozawa *et al*, "Statistical multiplexing gateway for piecewise CBR transmission channel", Packet Video'99, New York USA (1999)
16. L. Teixeira and M. Andrade, "Joint Control of MPEG VBR video over ATM networks", ICIP 97, Sta Barbara, CA, USA (1997)
17. M. Perkins and D. Arnstein, "Statistical Multiplexing of MPEG2 video channels", SMPTE 136th Technical Conference, LA, USA, (1994)
18. Compression Labs white paper, "The next step in Digital Broadcast: statistical multiplexer enhances compressed video MPEG2"
19. G. de Veciana, G. Kesidis and J. Walrand, "Resource Management in wide area ATM networks using effective bandwidth, IEEE JSAC, Vol. 13 (1995)
20. K. Ross, V. Veque, "Analytic Models for Separable Statistical Multiplexing", Univ of Pennsylvania, Dept of Systems Engineering (1994)
21. Didier Le Gall, "MPEG: A video compression standard for multimedia applications", Transactions of ACM (1991)
22. ISO/IEC 13818-2, ITU-T Rec H.262, "Generic coding of moving pictures and associated audio: video", (1994)

# The Study of Burst Admission Control in an Integrated Cellular System

Jie Zhou, Yoshikuni Onozato, and Ushio Yamamoto

Dept. of Computer Science, Faculty of Engineering, Gunma University  
1-5-1 Tenjin-cho, Kiryu, Gunma, 376-8515 Japan  
{zhou, onozato, kansuke}@nzt1.cs.gunma-u.ac.jp  
<http://www.nzt1.cs.gunma-u.ac.jp>

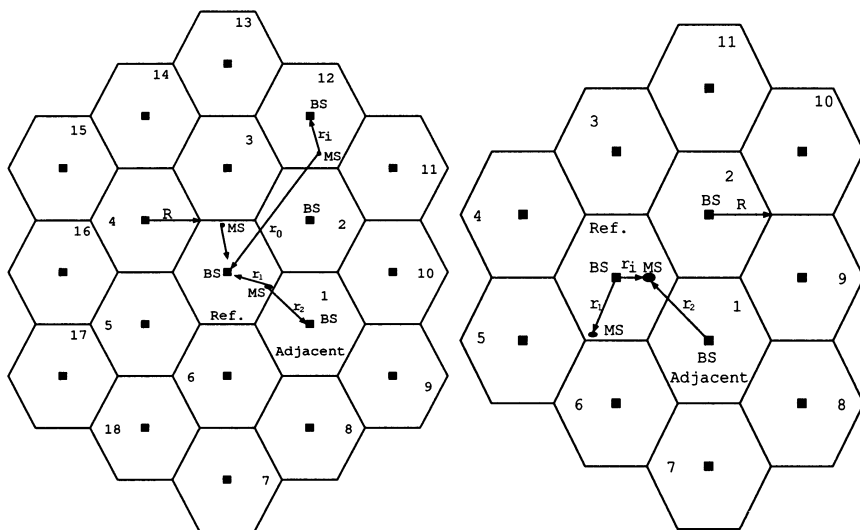
**Abstract.** CDMA has been proven to be one of the promising mainstream multiaccessing techniques in the third-generation mobile system. In this paper, we fully investigated the performance of a CDMA system which supports the data users. In the proposed algorithm[1][2], data user's burst admission is permitted based on the load situation of the system. Performance measures obtained included the system capacity, *ISR* limitation and the constraint set between the load of the system and the allowed maximum data rate. Based on the numerical results, we see the feasibility of integrated services supported in the CDMA system, where the burst admission is available, and what kinds of data services can be supported over the coverage area.

## 1 Introduction

Code Division Multiple Access (CDMA) has been a mainstream access technique in the third-generation mobile cellular system,[1][2]. The CDMA techniques have been reported to offer attractions such as high link capacity, more efficient spectral utilization, soft capacity and anti-multipath shadow fading[3][4]. In the future CDMA systems, a good variety of narrow-band and wide-band applications is expected to be supported in the same system with sharing the same spreading spectrum.

In this paper, we use the “equivalent load” to address the issues about providing data transmission in the CDMA cellular system shown in Fig.1. When the system wants to support the data rate services, the challenge is to design an admission control mechanism and estimate the impact of shadow fading on the data transmission in order to achieve efficient reuse of bandwidth and allow more users to obtain high data services without affecting the  $E_b/N_0$  for the other different services. Based on the algorithm, where the burst admission is allowed in the coverage area, we investigate how to control the data rate and give the complete and detailed numerical results, especially provide the numerical results about the forward link and its optimization of the distance-based power control scheme. At first, we summarize the analysis of the CDMA system as follows.

In a CDMA system, the quality of transmission is susceptible to the well-known near-far issue. Power control is exercised to reduce such effects. The work



**Fig. 1.** A typical example of the reverse and the forward links in standard cellular layout

in this paper is to be done under the assumption of perfect power control[8] in the reverse link and the distance-based power control scheme supported in Refs.[5] and [6] for the forward link.

On the forward link, we consider the cellular system model in Fig.1 and the distance-based power control scheme as

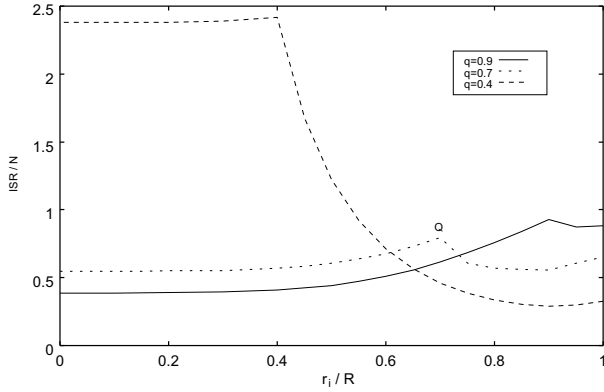
$$P(r_i) = \begin{cases} P_t \left( \frac{r_i}{R} \right)^\tau & \text{for } qR \leq r_i \leq R \\ P_t q^\tau & \text{for } 0 \leq r_i < qR \end{cases} \quad (1)$$

where  $r_i$  is the distance from the reference mobile station (MS) to its base station (BS).  $P_t$  is the forward link power required to reach those MS's located at the cell boundary.  $q$  is the power control parameter so that any users located at a distance less than  $qR$  is assured of receiving a minimum amount of transmitted power.

Based on the interference analysis[1][8][9] for the CDMA cellular system, we can easily obtain the mean of the total interference to the desired signal ratio  $I_t/S$  defined with  $ISR$ .  $ISR$  and variance of it, which is composed of intracellular and intercellular are given by[7]

$$\begin{aligned} ISR &= E[I_t/S] = E[I_{intra}/S] + E[I_{inter}/S] \\ &= E[I_{intra}/S] + \sum_{k=1}^K E[I(k, r_s)/S] \end{aligned} \quad (2)$$

$$V[I_t/S] = V[I_{inter}/S] = \sum_{k=1}^K V[I(k, r_s)/S] \quad (3)$$



**Fig. 2.** ISR/N versus distance  $r_i/R$  with the distance-based power control scheme[9] ( $\tau = 3$ ,  $\sigma = 8dB$ )

where,

$$E\left[\frac{I(k, r_s)}{S}\right] = \alpha \iint r_s^\mu F_1(r_s) \rho dA$$

$$V\left[\frac{I(k, r_s)}{S}\right] = \iint r_s^{2\mu} (\alpha H_1(r_s) - \alpha^2 F_1(r_s)) \rho dA$$

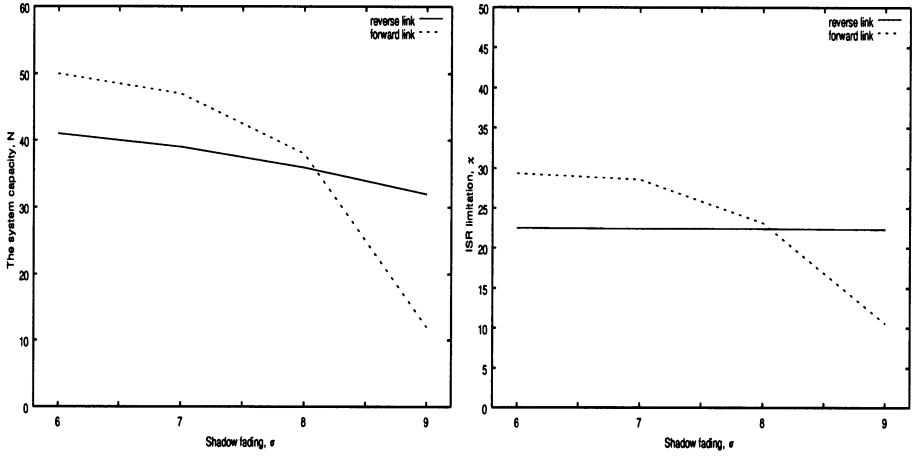
$$F_1(r_s) = \exp\left[(\sigma \ln 10 / 10)^2\right] \left\{ 1 - Q\left(\frac{10}{\sqrt{2}\sigma^2} \cdot \log\left(\frac{1}{r_s}\right)^\mu - \sqrt{2}\sigma^2 \frac{\ln 10}{10}\right) \right\}$$

$$H_1(r_s) = \exp\left[(\sigma \ln 10 / 5)^2\right] \left\{ 1 - Q\left(\frac{10}{\sqrt{2}\sigma^2} \cdot \log\left(\frac{1}{r_s}\right)^\mu - \sqrt{2}\sigma^2 \frac{\ln 10}{5}\right) \right\}$$

where  $Q(x) = \frac{1}{\sqrt{2\pi}} \int_x^\infty e^{-\frac{y^2}{2}} dy$ , and  $\rho$  is the density function of users distributed in the cell.  $r_s = r_i/r_0$  for the reverse link and  $r_s = r_i/r_2$  for the forward link shown in Fig.1.  $\mu$  is the path loss exponent.  $\sigma$  is the standard deviation of shadow fading.  $\alpha$  is the voice activity factor.

Power control strategy is not only used for decreasing interference to the other systems, but also for achieving the balance of the forward link user capacity. In the power control strategy used in this paper, it is important issue to choose the power control parameter for the highest possible user capacity and make  $ISR$  versus  $r_i/R$  curves shown in Fig.2 as flat as possible, that means[2][6][9]

$$\min_{0 \leq q \leq 1} \left\{ \max_{0 \leq r_i \leq 1} ISR \right\} \quad (4)$$



**Fig. 3.** The system capacity,  $N$  and the  $ISR$  limitation,  $\chi$  versus shadow fading,  $\sigma$  ( $W = 1.25MHz$ ,  $R_b = 8kbps$ ,  $\mu = 4$ ,  $\alpha = 3/8$ )

We must minimize the maximum of Eq.(4), i.e., it is the optimal issue of the forward link power control scheme. Eq.(4) was evaluated numerically for the forward link. The worst case is termed as 'hole' (Q) shown in Fig.2[6][9] in the presence of shadow fading which is assumed as the Gaussian random variable[1][7].

Outage probability  $P_{out}$  for the reference MS in the analysis of the forward link or  $P_{out}$  for the reference BS in the analysis of the reverse link is defined as the probability that the bit error ratio,  $BER$  exceeds the certain level, such as  $10^{-3}$  for the voice user. Adequate performance ( $BER \leq 10^{-3}$ ) is achievable on the reverse link with the required  $(E_b/N_0)_v = 7$  dB and the forward link, 5 dB[3][4]. Gilhousen[8] and Ref.[9] proposed models for estimating the reverse link and forward link capacity, respectively.  $P_{out}$  is given by

$$P_{out} = \sum_{k=1}^{N-1} \left\{ \binom{N-1}{k} \alpha^k (1-\alpha)^{N-k-1} \right\} EQ \left( \frac{\kappa - \psi k - E[I_{inter}/S]}{\sqrt{V[I_{inter}/S]}} \right) \quad (5)$$

In Eq.(5),  $Q(x) = \frac{1}{\sqrt{2\pi}} \int_x^\infty e^{-\frac{y^2}{2}} dy$ .  $\kappa = \frac{W}{R_b} \left( \frac{N_0}{E_b} \right)_v$ .  $\psi$  is the weight coefficient of intracellular interference. which is calculated for the reverse and forward links, respectively.  $R_b$  is the information rate of the voice user.  $W$  is the spread bandwidth of the system.  $N$  is the system capacity defined as the maximum number of active users in one cell when the outage probability is equal to 0.01[7][8][9]. So the maximum  $ISR$  limitation of the CDMA cellular system,  $\chi$  is

$$\chi = \alpha \psi (N-1) + \alpha \eta_1 N \quad (6)$$

where,  $\eta_1$  is the equivalent intercellular interference coefficient according to the calculation of  $E[I_{inter}/S]$ . The numerical results of the CDMA cellular system with the voice users under IS-95-B protocol are depicted in Fig.3.

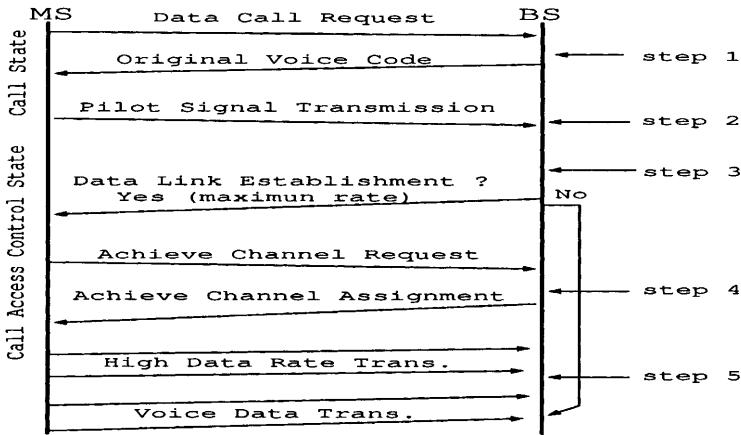


Fig. 4. Burst Admission Control Algorithm

In Section 2, the burst admission control algorithm is explained. The analytical methodologies are provided in Section 3. According to the burst admission control algorithm, we give the numerical results for the system designers in Section 4. Finally, the conclusions are given in Section 5.

2 Burst Admission Control Algorithm

The static analysis[3][10], concluded that the data rate permitted to the users who need data services is a function of its current path loss, shadow fading loss and the current load of the system. Because the load, traffic distribution and the current interference change dynamically due to the user mobility. The static analysis does not capture the relations among the voice users and effects of the data user due to the data user's burst admission, the user mobility and the load variation. Here, we summarize the burst admission control algorithm[1][2].

In the CDMA cellular system, to provide a smooth migration path to serve burst data traffic, we follow the admission control scheme that accounts for the interference, load and soft-handoff in making the access and *QOS* in the system, which protects voice isochronous users, but can accommodate peak rate user's admission as the interference and load on the assignment channel permits. It is important to derive a burst admission control algorithm in order to achieve the sharing of the same channel between the voice and data users, that both services are possible in the system without serious situations of outage probability and throughput occurred in the system.

In current CDMA cellular system employing mobile assisted soft handoff, BS provides the mobile with a neighbor list of pilot signal levels[1][2]. MS periodically detects the pilot signal levels and transmits them to its BS. These pilot

signal strengths may be used for determining the data user's admission as shown in Fig.4 which shows each voice user has a unique primary code. Then an active data user is assigned a fundamental code channel on the origination. According to all the pilot signal levels which are related to the interference, the load and the soft-handoff, its BS autonomously determines the maximum values of data rate or sets the user to maintain the origination. In particular, the data user will be granted a burst admission only if this burst admission will not affect the original voice users and data users.

In the burst admission algorithm, because the reuse in CDMA cellular system is based on the average interference, its calculation is important whether the load and interference-based demand assignment (LIDA)[1][2] and interference-based channel allocation (ICA)[11] schemes are used in the system. The burst admission from MS contains the data rate and required  $E_b/N_0$ . From the system management, the burst admission contains  $R_{max}$  and the area where the burst admission is allowed in the coverage area. They are important problems for system designers. For this purpose to achieve the most important step 3 of the burst admission control algorithm in Fig.4, we provide detail numerical results for the design of the reverse and forward links, respectively.

### 3 Analysis of Performance

#### 3.1 $R_{max}$ and the Constraint Set

In the CDMA cellular system, let us now consider the interference change when a data user is introduced in the CDMA system. The intracellular and intercellular interference are critical in determining  $R_{max}$ . The *ISR* constraint inequality for the reference cell is modified from Eq.(2), using "equivalent load"  $\phi(l)$  as

$$ISR = E[I_{intra}/S] + E[I_{inter}/S] + \phi(l) \leq \chi \quad (7)$$

where

$$\phi(l) = \frac{\left(\frac{E_b}{N_0}\right)_l / (P_{G_l} + \left(\frac{E_b}{N_0}\right)_l)}{\left(\frac{E_b}{N_0}\right)_v / (P_{G_v} + \left(\frac{E_b}{N_0}\right)_v)} \quad (8)$$

$\phi(l)$  is the "equivalent load" of one data user. It is considered that an active data user with data rate,  $R_l$  and required  $(E_b/I_0)_l$  consumes  $\phi(l)$  times greater equivalent resources of voice users if the data user is introduced in the system. In Eq.(8),  $P_{G_v} = W/R_b$  and  $P_{G_l} = W/R_l$ .

The *ISR* constraint inequality for the adjacent cell is

$$ISR = E[I_{intra}/S] + E[I_{inter}/S] + \phi(l)E[I_d(r_1, r_2, \sigma)] \leq \chi \quad (9)$$

where,  $E[I_d(r_1, r_2, \sigma)]$  is the normalized means of interference ratio between the path losses of  $r_1$  and  $r_2$  shown in Fig.1.

The data transmission is permitted to the users who require the data services depending on the locations of the user and the load of the system. According to



Eq.(9),  $R_{max}$  for the number of the voice users in the reference cell,  $N_v$  can be obtained as

$$R_{max} \leq \frac{W}{\left(\frac{E_b}{N_0}\right)_l} \cdot \frac{E[I_d(r_1, r_2, \sigma)] \cdot (P_{G_v} + \left(\frac{E_b}{N_0}\right)_v)}{(\chi - \alpha(\psi N_v + \eta_1 N)) \cdot \left(\frac{E_b}{N_0}\right)_v} - \left(\frac{E_b}{N_0}\right)_l \quad (10)$$

According to Eqs.(7) and (9), the *ISR* constraint inequality for the reference cell is

$$N_v \leq \frac{\chi - \alpha\eta_1 N - \phi(l)}{\alpha\psi} \quad (11)$$

and the *ISR* constraint inequality for the adjacent cell becomes

$$N_v \leq \frac{\chi - \alpha(\psi + \eta_2)N - \phi(l)E[I_d(r_1, r_2, \sigma)]}{\alpha(\eta_1 - \eta_2)}. \quad (12)$$

where  $\eta_2$  is the equivalent coefficient of total intercellular interference to the nearest adjacent cell, from all cells except for the reference cell as shown in Fig.1 for the reverse and forward links, respectively.

### 3.2 Reverse Link Assignment Analysis

Based on the standard propagation models[4][6], we obtain

$$E[I_d(r_1, r_2, \sigma)] = \left(\frac{r_1}{r_2}\right)^\mu F_1\left(\frac{r_1}{r_2}\right) \quad (13)$$

and  $V[I_d(r_1, r_2, \sigma)]$  as

$$V[I_d(r_1, r_2, \sigma)] = \left(\frac{r_1}{r_2}\right)^{2\mu} \left[ H_1\left(\frac{r_1}{r_2}\right) - F_1\left(\frac{r_1}{r_2}\right) \right] \quad (14)$$

### 3.3 Forward Link Assignment Analysis

Based on the distance-based power control scheme, the location of the data user is composed of two areas, such as,  $qR \leq r_1 \leq R$  and  $0 \leq r_1 < qR$ , respectively. When the data user is located in  $qR \leq r_1 \leq R$ , we obtain  $E[I_d(r_1, r_2, \sigma)]$  as[4][9]

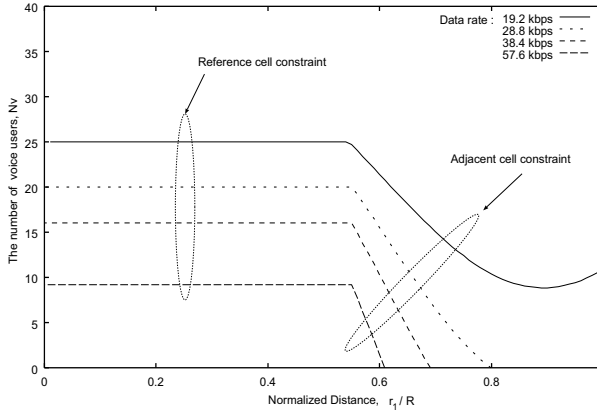
$$E[I_d(r_1, r_2, \sigma)] = \left(\frac{r_1}{R}\right)^\tau \frac{1}{q^\tau} r_s^\mu F_2(r_s) \quad (15)$$

and  $V[I_d(r_1, r_2, \sigma)]$  as

$$V[I_d(r_1, r_2, \sigma)] = \left(\frac{r_1}{R}\right)^{2\tau} \frac{1}{q^{2\tau}} r_s^{2\mu} \cdot \left[ H_2(r_s) - F_2(r_s) \right] \quad (16)$$

where in Eqs.(15) and (16), the explicit functions of  $F_2(r_s)$  and  $H_2(r_s)$  are expressed as

$$F_2(r_s) = \exp\left[(\sigma \ln 10 / 10)^2\right] \left\{ 1 - Q\left(\frac{10}{\sqrt{2}\sigma^2} \cdot \log\left(\frac{1}{r_s}\right)^\mu \left(\frac{qR}{r_1}\right)^\tau - \sqrt{2}\sigma^2 \frac{\ln 10}{10}\right) \right\}$$



**Fig. 5.**  $N_v$  versus the position of the data user,  $r_1/R$  about the reverse link ( $\sigma=8$  dB,  $\alpha=3/8$ ,  $\mu=4$ ,  $(\frac{E_b}{N_0})_v=7$  dB,  $(\frac{E_b}{N_0})_l=10$  dB)

and

$$H_2(r_s) = \exp\left[(\sigma \ln 10/5)^2\right] \left\{ 1 - Q\left(\frac{10}{\sqrt{2}\sigma^2} \cdot \log\left(\frac{1}{r_s}\right)^\mu \left(\frac{qR}{r_1}\right)^\tau - \sqrt{2}\sigma^2 \frac{\ln 10}{5}\right) \right\}$$

In the other case, when the data user is located in  $0 \leq r_1 < qR$ , we obtain  $E[I_d(r_1, r_2, \sigma)]$  as

$$E[I_d(r_1, r_2, \sigma)] = r_s^\mu F_1(r_s) \quad (17)$$

and  $V[I_d(r_1, r_2, \sigma)]$  as

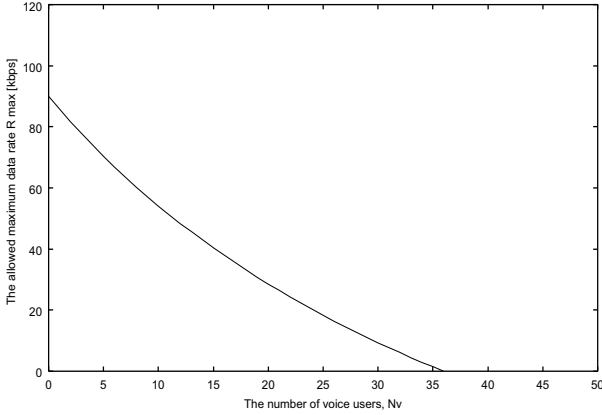
$$V[I_d(r_1, r_2, \sigma)] = r_s^{2\mu} [H_1(r_s) - F_1(r_s)] \quad (18)$$

For the forward link, we will focus our analysis on the reference MS located at the 'hole' [5][6][9].

## 4 Numerical Results and Discussions

### 4.1 System Capacity and $\chi$

Figure 3 shows the system capacity and  $\chi$  for the CDMA cellular system to various shadow fading environments. In Fig.3, we obtain the CDMA system capacity is equal to 36 users/cell for the reverse link, 38 users/cell under  $\sigma = 8$  dB. According to those results and Eq.(6), we can obtain the *ISR* limitation of the system as  $\chi = 22.39$  for the reverse link and  $\chi = 23.14$  for the forward link, respectively. We can also see the situations when  $\sigma$  is changed from 6 dB to 9 dB. The system capacity and  $\chi$  decrease rapidly for the forward link. That means the effect of the shadow fading is more significant on the forward link, which should be carefully designed and managed.



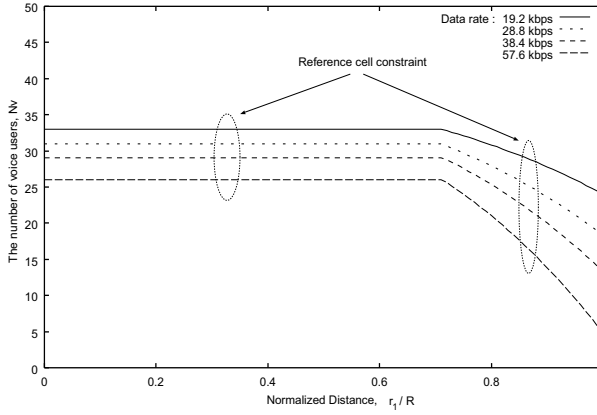
**Fig. 6.**  $R_{max}$  versus  $N_v$  about the reverse link ( $\sigma=8$  dB,  $\alpha=3/8$ ,  $\mu=4$ ,  $(\frac{E_b}{N_0})_v=7$  dB,  $(\frac{E_b}{N_0})_l=10$  dB)

## 4.2 Effects of Burst Admission and $R_{max}$

Figures 5 and 6 show the impact of one high data user's burst admission on the CDMA system reverse link. Notice that the curves in Fig.5 consist of two portions, i.e., the flat part and the rapid drop part. The first part is where the reference cell constraint dominates. The second part near the cell boundary is where the adjacent cell constraint becomes dominant. According to the figure, the important points are shown at near  $r_1/R = 0.55$ , we see if the data user with  $R_l$  is located at  $0 \leq r_1/R \leq 0.55$ ,  $N_v$  can be maintained as the same. when the data user is located at  $0.55 \leq r_1/R \leq 1$ , the locations of the data user greatly affect the system capacity, then should be managed carefully. From this part, the data services should not be permitted to any users when  $R_l \geq 38.4kbps$ . Based on the results, the high data transmission is available over a fraction of the cell coverage area as shown in Fig.5. It may be desirable to expand the coverage area for the high data transmission or at least to guarantee the minimum data rate.

We consider the user located at  $0 \leq r_1/R \leq 0.55$  who is desiring to obtain the data services as shown in Fig.5, where in Fig.6 we investigated  $R_{max}$  versus the load situation,  $N_v$ . It means keeping the value of  $N_v$ , we now determine  $R_{max}$  which may be allowed to any one user located in this area, who want to obtain the correspondent data services. For example, the numerical results show  $R_{max}$  with the required  $E_b/N_0 = 10dB$  can reach 90 kbps when  $N_v=0$ . When  $N_v=18$ , half load of the system,  $R_{max}$  reaches 35 kbps as  $\sigma = 8dB$ .

For the forward link, the reference cell constraint is dominant wherever the burst admission of the data user occurs in the reference cell shown in Fig.7. We also notice that the curves consist of two portions, i.e., the flat part and the smoothly decreasing part. The transmission of each data rate analyzed in this paper is available over whole cell coverage area, but it gives significant effects



**Fig. 7.**  $N_v$  based the reference cell constraint versus the position of the data user,  $r_1/R$  about the forward link ( $\sigma=8$  dB,  $\alpha=3/8$ ,  $\mu=4$ ,  $(\frac{E_b}{N_0})_v=5$  dB,  $(\frac{E_b}{N_0})_l=10$  dB,  $\tau=3$ ,  $q=0.7$ )

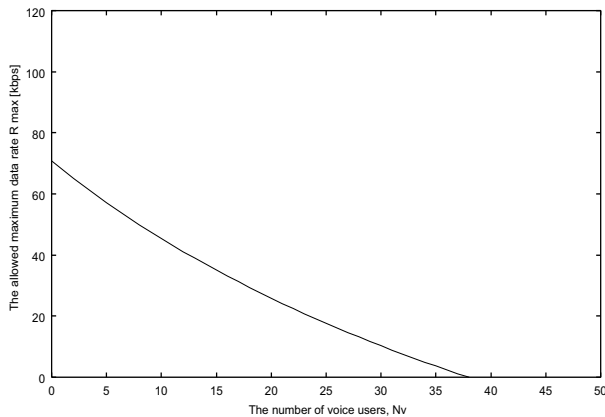
on the system when the data user is located in  $\frac{r_1}{R} \geq 0.7$ . It is also desirable to expand the coverage area for the high data rate transmission. When the user located in  $\frac{r_1}{R} \geq 0.7$  is to obtain high data rate services, the allowed data rate needs to be carefully controlled and managed.

When investigate  $R_{max}$  of the forward link, we assumed the data user is located at the boundary between cells. According to the distance-based power control scheme, its allocated transmitting power is greatest.  $R_{max}$  was estimated in Fig.8, in which  $R_{max}$  can reach 70 kbps when  $N_v=0$  and near 30 kbps as the load is in the half of the system for  $\sigma=8$  dB. When the data user moves from the boundary to its BS,  $R_{max}$  increases gradually.

## 5 Conclusion

In this paper, we have proposed a solution employing “equivalent load” to the performance of the CDMA cellular system with the integrated services in IS-95-B. Analysis is performed for the case when the burst admission of the data user occurs in the system. Especially, we estimated the characteristics of the forward link under the distance-based power control scheme and its optimal issue. Based on the numerical results, we can obtain our findings as follows:

1) Data users generate significant effects on the system because of its higher required  $E_b/N_0$  and its higher data rate, which means with lower processing gain in the same spreading bandwidth. If the data rate or the required  $E_b/N_0$  increases, the number of allowable voice users in the system decreases rapidly.



**Fig. 8.**  $R_{max}$  versus  $N_v$  about the forward link when the high rate data user is located at the boundary between cells ( $\sigma=8$  dB,  $\alpha=3/8$ ,  $\mu=4$ ,  $(\frac{E_b}{N_0})_v=5$  dB,  $(\frac{E_b}{N_0})_l=10$  dB),  $\tau = 3$ ,  $q = 0.7$ )

2) Based on the analytical results for the reverse and forward links, we suggest  $R_{max}$  is less than 50 kbps in the reverse link and 100 kbps in the forward link, because the near half area of the cell is not available for the data transmission.

3) For wide band CDMA systems of the third-generation cellular system all over the world in the future, our results can be directly proportional to the times of increased spread bandwidth.

4) The “equivalent load” proposed in this paper makes it possible to analyze various systems with a good variety of lower data rate, lower required  $E_b/N_0$  and higher data rate, higher required  $E_b/N_0$ .

Our studies demonstrate the feasibility of data users introduced in the CDMA cellular system. We have addressed several issues in some depth, such as  $R_{max}$  studies, admission method and the constraint set of the system. The analytical method depicted in this paper could be easily extended to treat much additional performance works such as outage probability and throughput because the complete formulas have been given in this paper.

## References

1. S.Kumar and S.Nanda, “High Data-Rate Packet Communications for Cellular Networks Using CDMA: Algorithms and Performance”, *IEEE J. Select. Areas Commun.*, Vol.17, No.3, pp.472-492, March 1999.
2. C-L.I and S.Nanda, “Load and Interference Based on Demand Assignment for Wireless CDMA network”, in *Proc. IEEE Globecom*, pp.235-241, 1996, Westminster, London.
3. H.G.Jeon, S.K.Kwon and C.E.Kang, “Reverse Link Capacity Analysis of a DS-SS-CDMA Cellular System with Mixed Rate Traffic”, *IEICE Trans. Commun.*, Vol.E81-B, No.6, pp1280-1284, June 1998.

4. J.S.Evans and D.Everitt, "On the Teletraffic Capacity of CDMA Cellular Networks", *IEEE Trans. on Veh. Technol.*, Vol.48, No.1, pp.135-165, Jan. 1999.
5. W.C.Y.Lee, "Overview of Cellular CDMA", *IEEE Trans. on Veh. Technol.*, Vol.40, No.2, pp.290-302, May 1991.
6. R.R.Gejji, "Forward-Link-Power Control in CDMA Cellular System", *IEEE Trans. on Veh. Technol.*, Vol.41, No.4, pp.532-536, Nov. 1992.
7. R.Prasad, *CDMA for Wireless Personal Communication*, Boston, MA: Artech, 1995.
8. K.S.Gilhousen, I.M.Jacobs, R.Padovani and A.J.Viterbi, L.A.Weaver, C.E.Wheatley, "On the Capacity of a Cellular CDMA System", *IEEE Trans. on Veh. Technol.*, Vol.40, No. 2, pp.303-311, May. 1991.
9. J.Zhou, Y.Onozato and U.Yamamoto, "Effect of Forward Link Power Control Strategy and Its Optimum Issue on Cellular Mobile System", submitted to *IEEE Trans. on Veh. Technol.*
10. J.Zhou, Y.Onozato and U.Yamamoto, "Throughput and Capacity Analysis of a DS-CDMA System with Integrated Services", Technical Report of IEICE, Vol.99, No.366, pp.1-6, Oct. 1999.
11. S.M.Shin, C.H.Cho and D.K.Sung "Interference-Based Channel Assignment for DS-CDMA Cellular System", *IEEE Trans. on Veh. Technol.*, Vol.48, No.1, pp.233-239, Jan. 1999.

# Model and optimal Call Admission Policy in Cellular Mobile Networks

Amine Berqia<sup>1</sup> and Noufissa Mikou<sup>1</sup>

<sup>1</sup>Université de Bourgogne  
UFR Sciences et Techniques  
LIRSI  
BP 47870

21078 Dijon Cedex, France  
{berqia,mikou}@crd.u-bourgogne.fr

**Abstract.** For current cellular networks, two important Quality of Service (*QoS*) measures are the fractions of new and handoff calls that are blocked due to unavailability of channels. Based on these *QoS* measures, we propose a queuing network model with impatient users for handoff and new calls in cellular mobile networks. The number of simultaneous calls, that can be supported, is modeled by  $C$  identical servers with exponentially distributed session duration for each one of them. Priority is given to handoffs over new calls. We use for that a Guard Channel policy that reserves a set of  $C_H$  channels for handoff calls, new calls being served at their arrival if there are more than  $C_H$  available channels. We derive loss probabilities, mean waiting times in the queues, and we show that the Guard Channel policy is optimal for minimizing a linear objective function of the new and handoff calls. Simulation results illustrating the behavior of our system are given.

## 1 Introduction

During the last few years there is a great demand for PCS (Personal Communication Services) which will provide reliable voice and data communications anytime and anywhere. The service area in a PCS network is partitioned into cells. In each cell, a Base Station (BS) is allocated a certain number of channels which is assigned to the mobile terminal MT enabling the MT to communicate with other MT's. As the MT moves from one cell to another, any active call needs to be allocated a channel in the destination cell. This event is called a handoff. Allocating radio resources to users with minimum blocking of new calls and dropping of handoff calls has become a vital issue in a mobile communication system design. It is known that dynamic channel allocation schemes can reduce these blocking and dropping probabilities for a reasonable range of loadings [2, 9]. However, allocating channels to every user whenever the sources are available may not be the optimal strategy in term of system performance.

The cellular network PCS have to be designed to guarantee the required Quality of Service (*QoS*) to each service or type of call (call meaning voice or data). Several policies and analysis are proposed in [4, 6] to minimize blocking

of handoff calls. In our paper we investigate a call admission policy which gives priority to handoff attempts over new call attempts.

In this paper, we assume that the cellular network is homogeneous and thus, we can examine a single network cell in isolation. The radio ports are modeled by  $C$  identical servers with exponentially distributed session duration for each one of them. We use a Guard Channel policy that reserves a set of  $C_H$  channels for handoff calls, new calls being served at their arrival if there are more than  $C_H$  available channels. Cellular networks performance was analysed in [6] using an approach where new and handoff calls were assumed to be Poisson arrival Processes. This approach is acceptable for macro cellular networks where a call might request for handoff only a few times. In this paper, we propose a Markov Modulated Poisson Process (MMPP) to describe voice handoffs (VH) arrival process because the arrival rate may depend on the state of the environment. The data Handoffs (DH) (as files transfer) and new calls (NC) arrive according to two different Poisson processes. Two separate finite buffer size queues are considered, one for each type of calls (handoff or new). The handoff calls enter service if there are available channels or join its queue if no idle channels are available and the queue is not full. Otherwise, They are lost. An arriving new call first checks the servers. It enters service if the number of busy servers is less than  $C - C_H$ , otherwise it enters the queue if the buffer is not full. Both handoff and new calls are impatient. That is a handoff leaves its queue and will be lost if it fails to get a channel before a Time Out (TO). And, in the same way, if a new user cannot get a channel before the end of the Patience Waiting Time (PWT), it leaves the queue. We derive loss probabilities, mean waiting times in both queues and we show that the Guard Channel policy is optimal for minimizing a linear objective function of the new and handoff calls. This paper is organized as follows: the queueing model is precisely defined in section 2. Loss probabilities and mean waiting time in queues are derived in section 3. In section 4, we show that the Guard Channel policy is optimal for minimizing a linear objective function of the new and handoff calls. Simulation results illustrating the behavior of our system are given and discussed in section 4. Finally, we conclude with some remarks in section 5.

## 2 The model

We consider a system with two separate queues for handoff and new calls with priority given to handoff over new attempts.

It is assumed that voice handoffs arrive according to an MMPP process  $A^{VH}$  with intensity  $\lambda(t) = \sum_{i=1}^n \lambda_i \times \mathbb{1}_{[Y(t)=i]}$  at time  $t$  where:

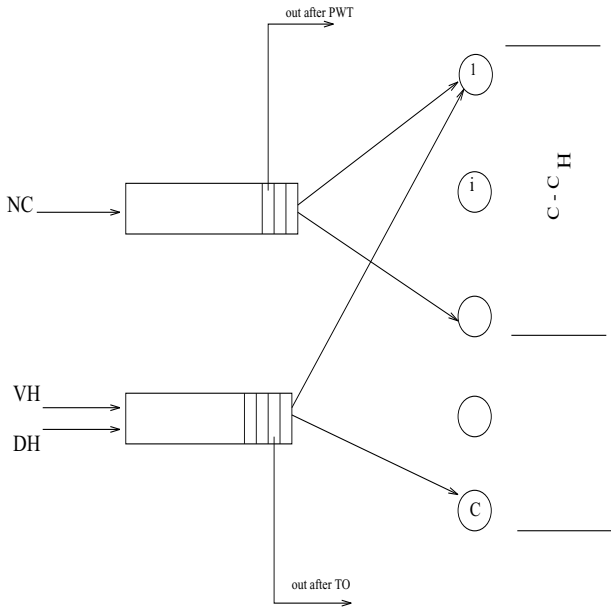
- $(Y(t))$  is an irreducible Markov chain with state space  $\{1, 2, \dots, n\}$  and an infinitesimal generator denoted by  $Q_n$ ;



- $\lambda_i, i = 1..n$ , are non-negative numbers;
- and

$$\text{– } \mathbb{1}_{[Y(t)=i]} = \begin{cases} 1 & \text{if } Y(t) = i \\ 0 & \text{otherwise.} \end{cases}$$

We suppose that the process of data handoff attempts is a Poisson process denoted by  $A^{DH}$  with arrival rate  $\lambda_{DH}$  and that mobile new users generate calls with a Poisson arrival denoted by  $A^{NC}$  with intensity  $\lambda_{NC}$ . The service is provided by  $C$  channels with exponentially distributed session duration for each one of them. We use a Guard Channel policy that reserves a set of  $C_H$  channels for handoff calls, new calls being served at their arrival if there are more than  $C_H$  available channels. The evolution of the system for both new



**Fig. 1.** The Model

- and handoff traffic flow is as follows :
- An arriving handoff call first checks the servers. It enters service if one of them is available. Otherwise, it is queued if the buffer is not full. A handoff in the queue will be lost if it fails to get a channel before a Time Out (TO).
  - An arriving new call first checks the servers. It enters service if the number of busy servers is less than  $C - C_H$ , otherwise it enters its queue if the buffer is not full. If a new user cannot get a channel before the end of the Patience Waiting

Time (PWT) it will be lost.

The Patience Waiting Time (PWT) and the Time Out (TO) are distributed exponentially with mean  $1/\mu_N$  and  $1/\mu_H$  respectively.

The queue buffer sizes are finite and denoted by  $K_H$  for handoff traffic and  $K_N$  for new calls.

Therefore, the new call generated by the mobile user calls may be lost at two cases: if the corresponding buffer is full at its arriving time or if its waiting time is greater than PWT. A handoff call may be lost if the corresponding buffer is full or if it fails to get a channel before that the TO expires.

The system is described by the multidimensional Markov Chain :

$$(X_{VH}(t), X_{DH}(t), X_{NC}(t), S(t), Y(t))$$

where :

- $X_{VH}(t)$  (respectively  $X_{DH}(t)$ ) is the number of voice Handoffs (respectively data handoffs) in their queue at time  $t$  respectively,
- $X_{NC}(t)$  is the number of new calls at time  $t$  in their queue,
- $S(t)$  is the number of busy servers at time  $t$ ,

and

- $Y(t)$  is the phase of the MMPP representing the state of the environment.

In this paper, we analyse our system in the steady state.

### 3 Performance analysis

Let  $B = \{(i, j, k, l, m) / 0 \leq i + j \leq K_H, 0 \leq k \leq K_N, 0 \leq l \leq C, 1 \leq m \leq n\}$  be the state space of the precedent multidimensional Markov Chain and  $P(i, j, k, l, m)$  be the stationary probability that the system is in the state  $(i, j, k, l, m)$ .

The  $P(i, j, k, l, m)$   $((i, j, k, l, m) \in B)$  verify the normalization condition :

$$\sum_{(i, j, k, l, m) \in B} P(i, j, k, l, m) = 1 \quad (1)$$

A VH or DH call is lost if the corresponding buffer is full or if its waiting time in the queue exceeds a Time Out (TO).

The probability that a VH mobile user attempt finds the buffer full is given by :

$$\tilde{P}_{VH} = \frac{\sum_{i+j=K_H} \sum_{k=0}^{K_N} \sum_{m=1}^n \lambda_m P(i, j, k, C, m)}{\sum_{m=1}^n \lambda_m \pi_m} \quad (2)$$

where  $\pi_Y = (\pi_1, \dots, \pi_n)$  is the unique vector probability solution of  $\pi_Y Q_n = 0$

(where  $Q_n$  is the infinitesimal generator of  $Y$ ).

A DH call finds the buffer full upon arrival with probability  $\tilde{P}_{DH}$  given by :

$$\tilde{P}_{DH} = \sum_{i+j=K_H} \sum_{k=0}^{K_N} \sum_{m=1}^n P(i, j, k, C, m) \quad (3)$$

If the time out expires before that a handoff gets a channel, the call is lost. The loss probability due to time out is given by :

$$P_{VTO} = \frac{\sum_{(i,j,k,l,m) \in B/i>0} i\mu_H P(i, j, k, l, m)}{(\sum_{m=1}^n \lambda_m \pi_m)(1 - \tilde{P}_{VH})} \quad (4)$$

for the voice handoffs

$$P_{DTO} = \frac{\sum_{(i,j,k,l,m) \in B/j>0} j\mu_H P(i, j, k, l, m)}{\lambda_{DH}(1 - \tilde{P}_{DH})} \quad (5)$$

for the data handoffs

So, the total loss probability for voice and data handoffs is respectively :

$$P_{VH} = \tilde{P}_{VH} + (1 - \tilde{P}_{VH})P_{VTO} \quad (6)$$

for the voice handoffs

$$P_{DH} = \tilde{P}_{DH} + (1 - \tilde{P}_{DH})P_{DTO} \quad (7)$$

for the data handoffs

In the same way, a new call is lost if at its arrival the corresponding buffer is full or if its waiting time in the queue exceeds the Patience Waiting Time (PWT).

The probability that a NC finds the buffer full is given by :

$$\tilde{P}_{NC} = \sum_{i+j=0}^{K_H} \sum_{m=1}^n \sum_{l=C_H}^C P(i, j, K_N, l, m) \quad (8)$$

The loss probability  $\tilde{P}_{NC}$  due to PWT is given by :

$$\tilde{P}_{PWT} = \frac{\sum_{(i,j,k,l,m) \in B/k>0} k\mu_N P(i, j, k, l, m)}{\lambda_{NC}(1 - \tilde{P}_{NC})} \quad (9)$$

So, the total loss probability for new calls is given by :

$$P_{NC} = \tilde{P}_{NC} + (1 - \tilde{P}_{NC})P_{PWT} \quad (10)$$

The mean number of VH in the queue is given by :

$$E(VH) = \sum_{(i,j,k,l,m) \in B} iP(i,j,k,l,m) \quad (11)$$

In the same way, we obtain the mean number of DH and NC in the queue :

$$E(DH) = \sum_{(i,j,k,l,m) \in B} jP(i,j,k,l,m) \quad (12)$$

$$E(NC) = \sum_{(i,j,k,l,m) \in B} kP(i,j,k,l,m) \quad (13)$$

and

$$E(DN) = \sum_{(i,j,k,l,m) \in B} lP(i,j,k,l,m) \quad (14)$$

The mean waiting time in the queue  $W$ , for voice handoffs, data handoffs and new calls can be obtained using Little's formula [1].

So, we have :

$$W(VH) = \frac{E(VH)}{\left( \sum_{m=1}^n \lambda_m \pi_m \right) (1 - \tilde{P}_{VH})} \quad (15)$$

$$W(DH) = \frac{E(DH)}{\lambda_{DH} (1 - \tilde{P}_{DH})} \quad (16)$$

and

$$W(NC) = \frac{E(NC)}{\lambda_{NC} (1 - \tilde{P}_{NC})} \quad (17)$$

The probabilities  $P(i,j,k,l,m)$  are numerically obtained using DNAmaca [5] which provides performance analysis sequence including steady state solution. In section 4, we prove that the Guard Channel policy is an optimal policy.

### 4 Optimal Admission Policy

In this section, we consider the problem of finding an optimal admission policy that determines the acceptance or rejection of new and handoff calls. For the cellular system described in section 3, let  $S(t)$  be the Markov chain representing the number of busy channels at time  $t$ . If, at steady state, that Markov chain is in state  $l$ , the policy consists then to choose an action  $a(l)$  in a set  $A$  of acceptance/reject actions depending on the state  $l$ , as follows : a call will be rejected due to impatience time or if there is no available channel and the buffer is full, otherwise it will be accepted. When the system is in state  $l$  and action  $a$  is chosen, then we incur a cost  $C(l, a)$  which will be  $L_h$  or  $L_n$  according to the action is a handoff or a new call reject. Moreover, the next state of the system is chosen according to the transition probabilities  $P_{lj}(a)$ . Our problem is to find a policy  $\pi^*$  over all call admission control policies  $\pi$  such that :

$$\phi_{\pi^*} = \min_{\pi} \phi_{\pi} \tag{18}$$

where

$$\phi_{\pi} = \lim_{N \rightarrow +\infty} E_{\pi} \big( \sum_{k=0}^{N-1} L_h \pi_{hk} + \sum_{k=0}^{N-1} L_n \pi_{nk} \big)$$

$E_{\pi}$  represents the conditional expectation given that the policy  $\pi$  is employed

and

the binary mapping  $\pi_{hk}, \pi_{nk}$  describing the actions for each state of the sytem are defined by :

$$\pi_{hk} = \begin{cases} 1 & \text{if } k\text{'th handoff is rejected} \\ 0 & \text{otherwise.} \end{cases}$$

$$\pi_{nk} = \begin{cases} 1 & \text{if } k\text{'th new call is rejected} \\ 0 & \text{otherwise.} \end{cases}$$

we need the following :

**Lemma 1.** *If there exists a bounded function  $h(l)$ ,  $l=0,1,...$ , and a constant  $g$  such that :*

$$g + h(l) = \min_a \{ C(l, a) + \sum_{j=0}^{\infty} P_{lj}(a) h(j) \}; l \geq 0 \tag{19}$$

*then there exists a stationary policy  $\pi^*$  such that :*

$$g = \phi_{\pi^*}(l) = \min_{\pi} \phi_{\pi}(l) \tag{20}$$

For the proof of lemma 2, see [8], page 144.

We obtain the following :

**Theorem 2.** *an optimal policy for the problem (18) is a Guard Channel policy.*

Now, we prove the existence of a bounded function  $h(l)$  and a constant  $g$  using the following construction:

For a discount factor  $\alpha \in (0, 1)$  and any policy  $\pi$ , we define :

$$V_\pi(l) = E_\pi \left[ \sum_{n=0}^{\infty} \alpha^n C(l_n, a_n) | l_0 = l \right], l \geq 0 \quad (21)$$

where  $E_\pi$  represents the conditional expectation given that the policy  $\pi$  is employed and the initial state is  $l$ .

Let,

$$V_\alpha(l) = \inf_\pi V_\pi(l) \quad (22)$$

A policy  $\pi^*$  is said to be  $\alpha$ -optimal if :

$$V_{\pi^*}(l) = V_\alpha(l) \forall l \geq 0 \quad (23)$$

For the proof of the theorem 2, we use the lemma 1.

We denote  $V_\alpha^k(l)$  the optimal cost for the  $k$ -stage starting in state  $l$ .

The  $\alpha$ -optimal cost function  $V_\alpha(\cdot)$  for  $0 \leq l \leq C$  satisfies :

$$\begin{aligned} V_\alpha^k(l) = & \\ & \lambda_h \min(V_\alpha^{k-1}(l+1), L_h + V_\alpha^{k-1}(l)) \\ & + \lambda_n \min(V_\alpha^{k-1}(l+1), L_n + V_\alpha^{k-1}(l)) \\ & + l\mu V_\alpha^{k-1}(l-1) + (C-l)\mu V_\alpha^{k-1}(l) \end{aligned}$$

Where  $\lambda_h = \lambda_{VH} + \lambda_{DH}$  and  $\lambda_n = \lambda_{NC}$ .

We consider that  $V_\alpha^k(C+1) = \infty$  and  $V_\alpha^k(-1) = V_\alpha^k(0)$ .

We define:

$$V_\alpha(l) = \lim_{N \rightarrow +\infty} V_\alpha^N(l) \quad (24)$$

It is shown in [10] that this function is non-decreasing and convex. Moreover, since the state space  $B$  of our system defined in section 3 is finite and the Markov chain induced by the control policy is irreducible,  $V_\alpha(l) - V_\alpha(0)$  is bounded uniformly according to [8], page 146.

Let us define :

$$h_\alpha(l) = V_\alpha(l) - V_\alpha(0) \quad (25)$$

to be the  $\alpha$ -cost of state  $l$  relative to state 0.

Then, from equations (24) and (25), we obtain :

$$\begin{aligned}
(1 - \alpha)V_\alpha(0) + h_\alpha(l) = \\
\lambda_h \min(h_\alpha(l+1), L_h + h_\alpha(l)) \\
+ \lambda_n \min(h_\alpha(l+1), L_n + h_\alpha(l)) \\
+ l\mu h_\alpha(l-1) + (C-l)\mu h_\alpha(l)
\end{aligned}$$

Now, for some sequence  $\alpha_n \rightarrow 1$ :

$h_{\alpha_n}$  converges to a function  $h(l)$  and  $(1 - \alpha_n)V_{\alpha_n}(0)$  converges to a constant  $g$ , then we obtain that :

$$\begin{aligned}
g + h(l) = \lambda_h \min(h(l+1), L_h + h(l)) \\
+ \lambda_n \min(h(l+1), L_n + h(l)) \\
+ l\mu h(l-1) + (C-l)\mu h(l)
\end{aligned}$$

Now, we are able to apply lemma 1 to say that a stationary policy  $a(l)$  verifying

$$a(l) = \phi_{\pi^*}(l) = \min_{\pi} \phi_{\pi}(l) \quad (26)$$

exists, and to give a:

## Proof of Theorem 2

The optimal policy  $a(l)$  that satisfies (26) is given by :

$$a(l) = \begin{cases} 1 & \text{if } h(l+1) - h(l) \leq L_n \\ 0 & \text{otherwise.} \end{cases}$$

Since  $h(l)$  is non-decreasing and convex,  $h(l+1) - h(l)$  is non-decreasing in  $l$ . Hence, there exist integers  $i_0$  and  $i_1$  where :

$$i_0 = \inf\{l : h(l+1) - h(l) > L_n\} \quad (27)$$

and

$$i_1 = \inf\{l : h(l+1) - h(l) > L_h\} \quad (28)$$

we note that  $i_1 > i_0$  since  $L_h > L_n$ .

So, the Guard channel policy that reserves  $C - i_0$  channels for handoff calls is the optimal policy for our problem.

5 Numerical and Simulation results

In this section, we present numerical and simulation results. For all experiments we consider  $C = 18$  servers with exponentially distributed session duration of parameter  $\mu = 1$ . Figures Fig.2 and Fig.3 represent loss probabilities versus Handoff and new calls buffer size respectively. In both cases, the parameters  $C_H$ , PWT and TO are respectively 5, 40 and 30. When the buffer size increases, we note that the loss probability of both Handoff and new calls decreases, and the handoffs loss probability is lower. Figures Fig.4 and Fig.5 depict loss probabilities of handoff

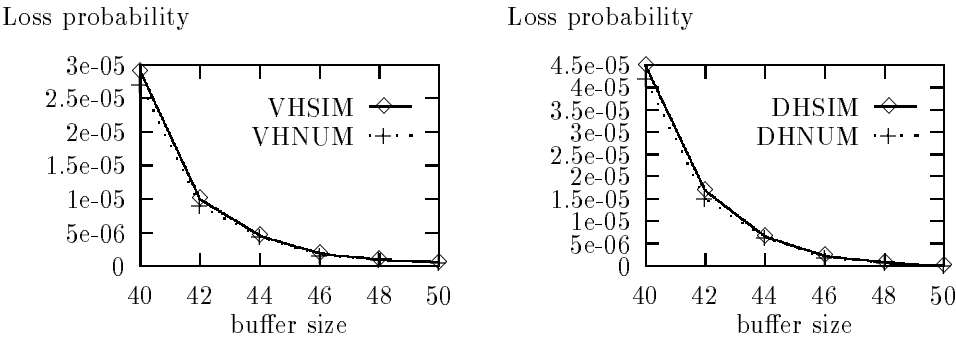


Fig. 2. handoffs loss probabilities vs. buffer size,  $C = 18, C_H = 5, PWT = 40, TO = 30$

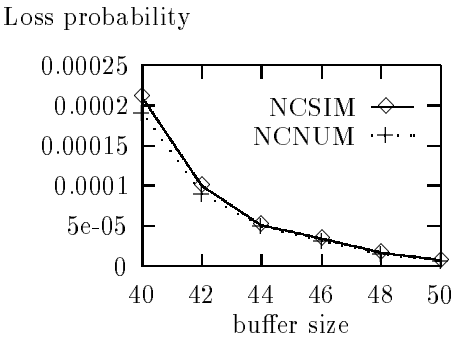


Fig. 3. new call loss probabilities vs. buffers size,  $C = 18, C_H = 5, PWT = 40, TO = 30$

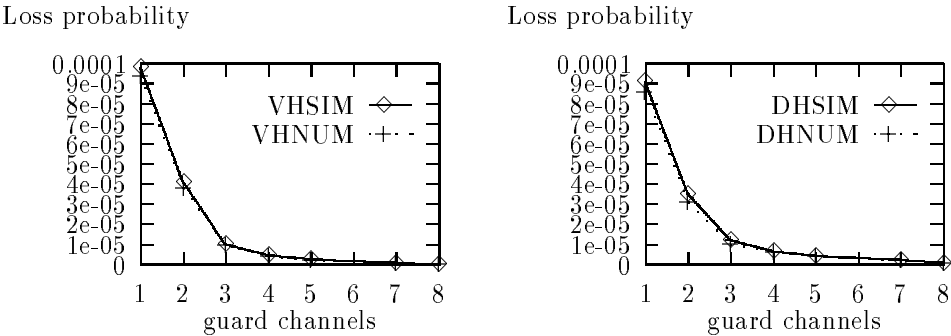
and new calls versus number of guard channels  $C_H$ . Loss probability decreases



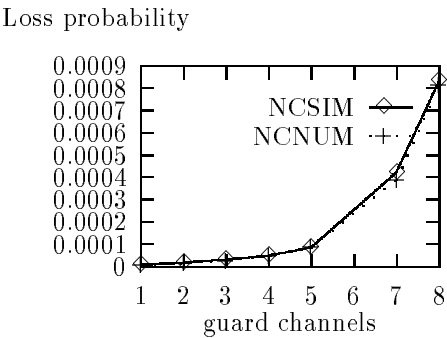
for Handoff mobile users when we increase the number of guard channels. The Loss probability of new calls increases when we increase the number of guard channels.

In Figures Fig.4 and Fig.5 we see that if we reserve 3 or 4 channels from 18 channels, the Handoff loss probability is lower than  $10^{-6}$ , and the new call loss probability does not exceed the value  $10^{-4}$ . So, we propose to reserve 20% of channels to Handoffs.

The effect of channel reservation on the response times is depicted in Fig.6.



**Fig. 4.** handoffs loss probabilities vs.guard channels,  $C = 18, PWT = 40, TO = 30$



**Fig. 5.** new call loss probabilities vs.guard channels,  $C = 18, PWT = 40, TO = 30$

response time

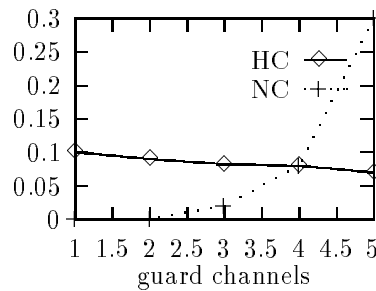


Fig. 6. response time vs. number of guard channels

## 6 Conclusion

In this paper, we have considered a Guard channel policy that gives priority to handoff over new calls in a cellular mobile network. We show that this policy is optimal for a linear objective function of the new and handoff call blocking probabilities. Under this policy, performance for handoffs are optimal without serious degradation for new calls. Moreover, blocking probabilities and response times for both types of calls are obtained and numerical results show the impact of the buffer size and the number of guard channels on the system behaviour. Finally Simulation results are conducted to validate our results.

## References

1. F. Bacceli and P. Brémaud, *Elements of Queueing Theory*, Springer Verlag, Series Applications, August 1993.
2. R. Beck and H. Panzer, *Strategies for handover and dynamic channel allocation in micro-cellular mobile radio systems*, Vehicular Technology Conference, 1989.
3. D. P. Bertsekas, *Dynamic Programming : Deterministic and Stochastic models*, Prentice Hall, Englewood Cliffs, NJ, 1987.
4. R. Guérin, *Queueing Blocking System with Two Arrival Streams and Guard Channels*, IEEE Trans. on Communication, vol. 36, no.2, pp. 153-163, Feb. 1988.
5. W. J. Knottenbelt, *DNAmaca: A Generalized Performance Analyser for Timed Transition Systems*, Master's Thesis, University of Cape Town, South Africa, 1995.
6. D. W. Mc Millan, *Delay Analysis of a Cellular Mobile Priority Queueing System*, IEEE/ ACM Trans. on Networking vol. 3, no. 3, pp. 310-319, Jun. 1995.
7. I. Mitran, *Probabilistic Modeling*, Cambridge University PRESS 1998.
8. S. Ross, *Applied Probability Models with Optimization Applications*, Holden-Day, 1970.
9. M. Zhang and T. S. P. Yum, *Comparisons of channel assignment strategies in cellular mobile telephone systems*, IEEE Transactions on Vehicular Technology, 38(4), November 1989.
10. Jean Walrand, *An introduction to queueing networks*, Prentice-Hall, Englewood Cliffs, NH, 1988.

# Measurement-based Pre-Assignment Scheme with Connection-level QoS Support for Multiservice Mobile Networks

Xiaoyuan Luo<sup>1</sup>, Ian Thng<sup>2</sup>, Bo Li<sup>3</sup> and Shengming Jiang<sup>3</sup>

<sup>1,2</sup> Department of Electrical Engineering

<sup>4</sup> Centre for Wireless Communication

National University of Singapore

E-mail: {engp7708,eletlj,cwcjsm}@nus.edu.sg

<sup>3</sup>Department of Computer Science

Hong Kong University of Science and Technology

E-mail: bli@cs.ust.hk

## Abstract

This paper contributes a new adaptive approach to prevent handoff failure due to lack of resources in adjacent cells in a mobile cellular network. Known as the Measurement-based Pre-assignment (MPr) technique, the technique is useful for implementation in micro/pico cellular networks which offers a Quality of Service (QoS) guarantee against call dropping. The technique is fully automated, simple, efficient, robust and generic for implementation. The notable feature is that the technique accepts different QoS specifications for different grades of handoff calls and ensures that these QoS levels are preserved under changing load conditions.

**Subject Area:** Handoff Support, Quality of Service, Cellular Networks

## 1 Introduction

The event of an unfinished call being terminated by the system is known to infuriate users much more compared to a new call that is unable to get through. In fact, one such experience is good enough to nullify any good opinions the user might have on a number of other previous calls that was successfully made. These rude terminations often occur at cell boundaries when a handoff request is being made to a cell that has insufficient resources. With cell sizes being systematically decreased to promote better frequency reuse which, by the same token, also increases handoff events, mechanisms that ensure that the probability of handoff dropping events does not exceed a specified QoS level is certainly useful to micro-cell wireless networks [1].

To reduce handoff failure due to lack of resources in adjacent cells, a basic approach is to reserve resources for handoffs in each cell. The well-known guard channel (GC) scheme and its variations [2, 3, 4] reserve a fixed number of channels in each cell exclusively for handoffs in a single service environment. Generally, GC schemes are not adaptive to changing load conditions and no QoS guarantee is provided. A recent work [5] has shown that the guard channel scheme is unable to provide fair handoff support to different service types, since more bandwidth has to be reserved for wide bandwidth calls.

A variety of channel allocation strategies have been contributed for multi-service support in mobile networks. Based on multi-dimensional birth-death processes, the author in [5] developed

a framework that is useful for teletraffic performance analysis and modelling for a broad class of problems stemming from cellular communication architecture. In [6, 7], two-tier resource allocation schemes for two types of incoming new calls were considered, a narrow-band type and a wide-band type. Complete sharing (CS), complete partitioning (CP), and hybrid allocation were investigated. But both schemes assume that only the narrow-band call can request handoffs. In [8], resource management for an integrated mobile networks with real-time connections and non-real-time connections is broken up into two independent issues, namely, class-based wireless call admission control, and bandwidth allocation to admitted connections according to their quality of service requirement. However, non-real-time connections are assumed not to request for handoff and real-time connections are assumed to have the same bandwidth requirements. This limits the scope of the models.

The schemes surveyed above have focused more on channel allocation as opposed to handoff support which is more relevant in this paper. Existing techniques [9]-[17] utilise a combination of motion detection and load status broadcasting (to neighbouring cells) to support multi-service handoffs. Handoff support schemes using motion prediction models can be found in [9, 10, 11]. The shadow cluster concept [12], and its event-based variation [13], exploit the network capability to predict the motion probability of every mobile. The servicing base station must maintain status of each active call, and delivery it to all cells in the shadow cluster, i.e. the cells that the mobile might visit in the future. The efficiency of these schemes depends heavily on the accuracy of the network in predicting the potential movement of mobile hosts. The use of load status or number of active channels in neighbouring cells, to a determine local admission thresholds, are found in [15, 14, 17], etc. These schemes create dependencies on neighboring cells for local call admission process, which may not do too well on the issue of robustness, for example, the possible event of a neighbouring cell who are no longer compliant to the information exchange process due to some unforeseen circumstances. Three problems remain unanswered in supporting handoff calls in a multi-service environment:

- Implementation complexity. Most schemes assume additional function to be performed by the network (e.g. signalling and motion prediction), and require base station to have high computation capability and large buffers to store call status or load status in neighbouring cells.
- The assignment of reserved channels is neglected. Most contributions have been focused on determining the total amount of resources to be reserved for all handoff calls. It is not clear which of the reserved channels are for which class of calls. If reserved resources are not distinguished for different call classes, wide-band calls will encounter much higher handoff dropping probabilities than narrow-band calls, as pointed out in [5] and illustrated later with numerical results.
- Complicated analytical model. Due to the modelling complexity of the multi-service system, a simple close-form formula to relate the desired QoS with the amount of resource to reserve is currently unavailable.

In this paper, we propose a Measurement Pre-reservation (MPR) scheme for the support of multi-service handoff with QoS guarantee on handoff dropping. The MPR scheme is adaptive to changing load conditions, changing amount of reserved resources dynamically. The reserved resources for each type of call with different QoS requirement is differentiated. A close-form solution relates the specified QoS levels with the amount of channels to reserve for each class of calls. This gives the wireless network provider full and easy control of the system even to the extent of changing the QoS level of any call type under widely varying load scenarios. Also, the MPR scheme preserves cell independence. It is a generic scheme that can be implemented in any cell even the adjacent cell is not compliant to the proposed scheme. It does not maintain the motion status for each call and there is no requirement for information exchanges amongst neighbouring cells. Consequently, issues leading to inefficiency of the system like late or delayed arrivals of inter-cell information

cannot affect an MPr-compliant cell. Finally, the MPr scheme is practical for implementation as the operation for a base station is fairly simple.

This paper is organised as follows. We start from illustration of basic measurement-based pre-reservation for handoff a single service environment. Thereafter, we present the enhancement to support multiple QoS levels for different type of handoff calls in a multi-service environment. Finally, simulation results will be presented to illustrate the usefulness of the technique.

## 2 The Pre-Assignment Methodology in MPr

A fundamental difference between the GC scheme and the MPr scheme is the methodology of assigning the reserved channels. Figure 1 and 2 illustrates the resource management of the proposed MPr scheme, in comparison with the Guard Channel scheme, in a homogeneous call environment. The GC scheme employs a post-assignment methodology as illustrated in Figure 1 where handoff calls and new calls are first assigned to the common channels. In the event the common channels are depleted, new calls are blocked while handoff calls are assigned to the reserved channels. In the pre-assignment methodology of Figure 2 that is commensurate with the MPr technique, handoff calls are immediately assigned to reserved channels. If the reserved channels are depleted, handoff calls will have to compete with new calls for the common channels.

By using the pre-reservation methodology, the MPr scheme enjoys a desirable feature that a stated QoS can be easily “understood” and realised by the resource controller. The MPr scheme computes a reservation pool size for the stated QoS specification under the given loading conditions. It would also be clear later, that a smooth enhancement of the MPr scheme is found to support multiple QoS specification in a multi-service environment. In contrast, for a stated QoS, the number of Guard Channels to be reserved is cumbersome to determine, as it relies on state-transition queuing analysis as used in [2, 3] or looking up multiple tables with different loading parameters. Although the GC scheme is optimal for a evaluation function constructed by using the linear sum of call blocking probability and handoff dropping probability [14], the analytical complexity may not be desirable to guarantee a stated QoS under changing load conditions. For the same reason, extending the guard channel scheme for handoff support a multi-service environment will encounter considerable difficulty [5].

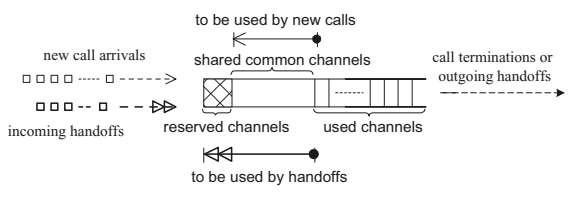


Figure 1: Post-assignment in a Guard Channel scheme

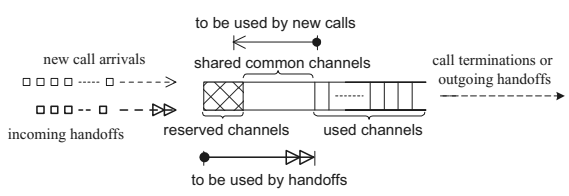


Figure 2: Pre-assignment in the proposed MPr scheme

Two mechanisms are required for the MPr scheme. The first mechanism is described as follows: Given a target reservation pool size, i.e.  $R_{RPS}$ , which provides the required call dropping QoS specified for the service, a mechanism is required to maintain the number of reserved channels at time  $t$ ,  $R(t)$ , towards  $R_{RPS}$ .

As an illustration, consider the arrival of a handoff call at time  $t$ . Due to the pre-assignment principle, the number of reserved channels will immediately shrink by one as follows:

$$R(t^+) = R(t^-) - 1 < R_{RPS} \quad (1)$$

The second mechanism is, to determine  $R_{RPS}$  in order to satisfy the desired call dropping QoS for the cell. The design of the second mechanism is rather challenging as it is constrained by the following factors:

- The mechanism is adaptive towards changing load conditions, shrinking/enlarging where necessary to ensure the specified call dropping QoS for the cell.
- The mechanism can be deployed in the network in a distributed fashion where the mechanisms run independently of each other from cell to cell.
- For generic and robust implementation, the mechanism should not require information exchanges amongst neighbouring cells.
- The mechanism should also support multiple QoS requirements for different types of services.

In the following section, we present the two mechanisms required to perform MPr in a wireless cellular network that is operating in a homogeneous call environment.

### 3 MPr for a Homogeneous Call Environment

A typical homogeneous service environment often assumes that each call, i.e. handoff call or new call, occupies one unit of bandwidth, or a channel. It is also assumed that new call arrivals and handoff call arrivals are Poisson processes [3, 4]. In addition, the call life time and call dwelling time within a cell are assumed to be exponentially distributed.

#### 3.1 Token-property Update of $R(t)$

As mentioned before, a mechanism is required to update  $R(t)$  (the number of reserved channels at time  $t$ ) towards a given  $R_{RPS}$  (the reservation pool size). Figure 3 illustrates the update logic being adopted for the MPr algorithm.

Figure 3 illustrates a very important property of the MPr scheme which we refer to as the *token-property update*. It is described as follows:

A channel can only be reserved at time  $t$  if the following three conditions are true.

- At time  $t^-$ , the channel was in the busy state, servicing a call.
- At time  $t^-$ ,  $R(t^-) < R_{RPS}$ .
- At time  $t$ , the above channel is released by the call due to termination or handoff.

The token-property update, which will be more meaningful later, also indicates the following:

- A channel that has been designated to the common pool cannot transit directly to the reservation pool. It has to transit a busy state first before it can be considered for designation in the reservation pool. See Figure 3.

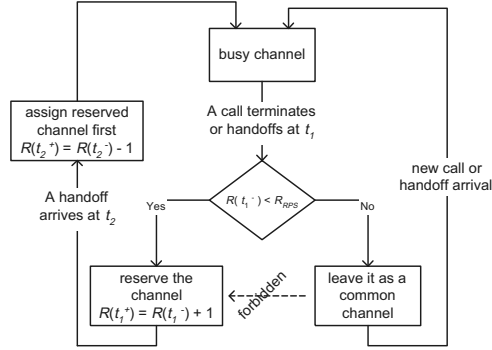


Figure 3: State transition of a radio channel

- If the number of reserved channels have not reached the desired the number, i.e.  $R(t^-) < R_{RPS}$ , then a channel returned to the system, when a call terminates or handoffs, is immediately reserved.
- If the reservation pool is depleted, i.e.  $R(t) = 0$ , then a handoff call at time  $t$  will have to compete with new calls for an idle channel in the common pool. A strict dropping policy may be employed, which drops a handoff request under such condition, but is not encouraged for the same reasons as discussed in Section 1. Comparison of such a variation with the normal admission condition for handoff calls will be presented in the numerical section.

The token-property update policy holds the key to the simplicity of the MPr scheme and contributes, to a certain extent, some of the favourable features of the MPr scheme. This will be elaborated further in the next section.

### 3.2 Update of $R_{RPS}$

With the token-bucket update policy of  $R(t)$  in place, it is now rather easy to develop a token bucket model for updating  $R_{RPS}$  as illustrated in Figure 4. In the figure,

- The size of the token bucket represents the  $R_{RPS}$ , i.e. the reservation pool size.
- Arriving tokens at the bucket ingress represent events where channels are released (by call handoff or call termination).
- If the bucket is full, i.e.  $R(t) = R_{RPS}$ , then any arriving tokens are discarded, meaning that a released channel is designated to be a common channel.
- Arriving handoff calls are assigned reserved channels if tokens are available. If the token bucket is empty of tokens (i.e. reservation pool is depleted), then there is no guarantee the handoff call will be admitted as it depends on the availability of common pool channels.

The token bucket is essentially a M/M/1 queue system with finite capacity  $R_{RPS}$ . We shall see that the possibility for a handoff to be accepted when there is no channel reserved is fairly low in the proposed MPr scheme, although it is possible. Such case requires some conditions to be satisfied. Firstly, the free channel must be released at the time the reservation pool was full so that it was not captured. Another necessary condition is that during the period handoff calls move in and use up all the reserved channels, no new call attempts within the radio cell utilise that particular free

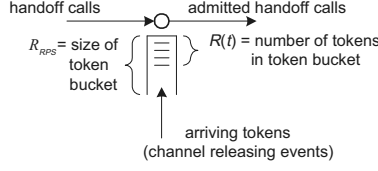


Figure 4: The analytical model of the MPr scheme

channel. Therefore, the probability that the reservation pool is empty effectively gives a tight upper bound for call dropping probability.

If the call dropping QoS probability is specified to be  $P_{QoS}$ , then the MPr scheme has to determine a suitable token bucket size to ensure that

$$P_{QoS} = P(\text{handoff call is dropped}) \leq P(\text{token bucket is empty}) \quad (2)$$

Employing the M/M/1/k queue principles [18], we note that

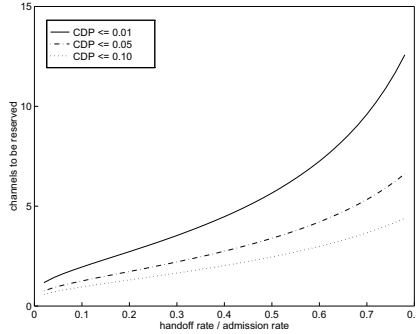
$$P(\text{token bucket is empty}) = \frac{1 - \rho}{1 - \rho^{R_{RPS} - 1}} \quad (3)$$

$$\text{where } \rho = \frac{\text{channel releasing rate}}{\text{incoming handoff rate}} = \frac{\mu_r}{\lambda_h} \quad (4)$$

Incorporating the above equations, we obtain the update condition for minimum  $R_{RPS}$  as follows,

$$R_{RPS} \geq \frac{\ln \frac{P_{QoS} + \rho - 1}{P_{QoS}}}{\ln \rho} - 1 \quad (5)$$

It is noted that the update for  $R_{RPS}$  in 5 only requires local cell metrics  $\mu_r$  and  $\lambda_h$  - this is precisely the strength of the MPr scheme in terms of robustness and generic deployment in a wireless network where not all cells are MPr compliant.


 Figure 5: Channels to be reserved over  $1/\rho$ 

### 3.3 Fine Tuning of the $R_{RPS}$ Update

There are a number of issues regarding the update equation of  $R_{RPS}$  in 5, the first being the estimate of the channel releasing rate  $\mu_r$  which can be replaced with

$$\mu_r = \lambda_a = \text{successful call admittance rate} \quad (6)$$



Therefore there is no necessity to incorporate a MPrarate device to monitor the channel releasing rate as the call admission control device of the cell can indirectly perform that function.

It is now clear that the  $1/\rho$  represents the percentage of incoming handoff calls in the total successful call attempts, provided handoff failure is negligible because of the reserved resources. By monitoring  $\rho$  in 5,  $R_{RPS}$  can be adaptively adjusted. This is seen more clearly in Figure 5 which plots  $R_{RPS}$  against  $1/\rho$  for various  $P_{QoS}$ . Notice that as the percentage of handoff rate increases, so does the number of channels to be reserved. The more stringent the QoS requirement, the more channels will be reserved.

In addition, there is no stringent requirement on the refresh frequency of  $\lambda_a$  and  $\lambda_h$ , neither the identical update period for every cell in [15], nor the periodical flooding of call status to neighboring cells as in [13, 13]. Once every few minutes should be adequate. Chances are that channels are reserved but handoff attempts decrease afterwards, e.g., when attendants are seated and a conference starts, the arrival rate would decrease drastically. In such a case, excessively reserved channels will be released upon the above mentioned measurement is refreshed and new size of the reservation pool is re-computed. To smooth out random or drastic measurement inaccuracies, an exponential smoothing average formulae can be employed to obtain a moving average  $\lambda_{ave}$  as follows:

$$\lambda_{ave}(n) = \alpha \lambda_{ave}(n-1) + (1-\alpha) \lambda_{new} \quad (7)$$

each time a new rate  $\lambda_{new}$  is measured.

## 4 MPr Support for Multi-Service Handoff Calls

In a multi-service environment with mixed call types, either of the following reasons require the cell to differentiate handoffs belonging to different call types: *i*) Different calls  $i$  have different QoS requirement in terms of handoff dropping, i.e.  $P_{QoS_i}$ . *ii*) Different calls  $i$  have different channel requirements  $B_i$ .

The MPr algorithm is amenable for application in both a complete partitioning environment and a resource sharing environment. Other bandwidth partitioning strategies that allocate bandwidth fairly for different traffic classes while maximising network throughput can be found in [8, 19].

### 4.1 Multi-Service with Complete Partitioning Resource Allocation

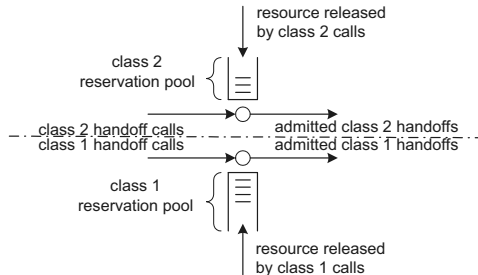


Figure 6: A complete partitioning multi-service environment

With complete partitioning (CP) resource allocation, the allocated bandwidth boundary for each service class is clearly demarcated. Thus an incoming class  $i$  call consumes bandwidth allocated for class  $i$  and an outgoing class  $i$  call returns the resources back to its class. Consequently, the handoff reservation pool for each class is also separated from each other and belongs strictly to each class. In this environment, independent MPr mechanisms outlined in Section 3 can be used to

update the required reservation pool for each class as shown in Figure 6. Thus the MPr mechanism in this scenario is just a trivial extension of the one in the homogenous call environment. A point to note is that 5 should take into consideration the different amount of bandwidth each class of call requires. Thus Equation 5 should be modified to

$$R_{RPS_i} \geq B_i \times \left( \frac{\ln \frac{P_{QoS_i} + \rho_i - 1}{P_{QoS_i}}}{\ln \rho_i} - 1 \right) \quad (8)$$

where  $R_{RPS_i}$  is the number of channels to be reserved in class  $i$  and

$$\begin{aligned} \rho_i &= \frac{\mu_{ri}}{\lambda_{hi}} = \frac{\text{call releasing rate for class } i}{\text{incoming handoff rate for class } i} \\ &= \frac{\lambda_{ai}}{\lambda_{hi}} = \frac{\text{successful call admittance rate for class } i}{\text{incoming handoff rate for class } i} \end{aligned} \quad (9)$$

## 4.2 Multi-Service Resource Sharing Environment

In a resource sharing multi-service environment, the bandwidth of the cell is not demarcated and is shared by different call types. In order to support the different handoff requirements, a total of  $i$  independent MPr processes is required, each maintaining their own respective reservation pools for each class  $i$  as illustrated in Figure 7. An incoming class  $i$  handoff call will immediately be assigned reserved channels corresponding to its class. And as before, whichever released channels not reserved (i.e. the rightful token buffer is full) merely becomes part of the pool of common channels.

However, a difficulty is now to decide which token buffer in Figure 7 deserves the arriving tokens (arising out of resource releasing events). Specifically, the difficulty lies with characterising the overall channel releasing process and designing an appropriate scheduling mechanism that preserves the M/M/1/k model for each reservation pool class. The difficulty can be overcome by the use of a poorer performing fictitious model rather than the actual model. Then based on the fictitious model, an appropriate and simple dispatching mechanism is described which is also applicable for use in the original model (which is better performing). Finally the appropriate update equations for the various reservation pool classes of the original model are provided.

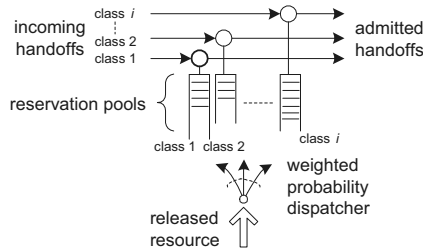


Figure 7: A resource sharing multi-service environment

### 4.2.1 The Overall Channel Releasing Process

The overall channel releasing process associated with Figure 7 is not a conventional Poisson process but a sum of Poisson processes with bulk arrivals. The class  $i$  call completion or outgoing-handoff process would return resource to the system at the rate of  $\mu_{ri}$  with bulk arrival  $B_i$ , where  $B_i$  is units of bandwidth used by class  $i$  calls. Characterising the overall channel releasing process is,

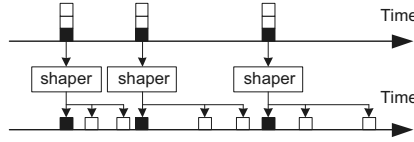


Figure 8: Shaping the bulk arrival into a more inferior system

in general, rather difficult, but the difficulty can be overcome by considering a inferior fictitious system.

Due to the memory-less nature of the Poisson process, a conventional Poisson process at the rate  $\mu'_{ri}$  can be constructed from a given bulk arrival Poisson process with rate  $\mu_{ri}$  and bulk size  $B_i$  via shapers, which distribute/delay exponentially (in time) the bulk arriving channels, as illustrated in Figure 8. If we subject all the bulk arrival channel releasing processes of the cell to be exponentially shaped to conventional Poisson processes, then the overall channel releasing process of the fictitious model is a sum of conventional Poisson processes. Consequently, the overall channel releasing process of the fictitious model is a well characterised conventional Poisson process with rate  $\mu_r$  given as follows:

$$\mu'_r = \sum_i \mu_{ri} B_i = \sum_i \lambda_{ai} B_i \quad (10)$$

where  $\lambda_{ai}$  is defined in 6.

The model is fictitious since, in order to distribute the bulk arrivals to exactly construct a conventional Poisson process, shapers ought to have prior knowledge of future bulk arrival events - which is impractical. Nonetheless, the fictitious domain is useful for further analysis, not only because its overall channel releasing process is well characterised, but it is noted that in the fictitious system, its channel reservation process is inferior compared to the actual system's reservation process since shapers in the fictitious system introduce artificial delays to arriving resources that can otherwise be immediately reserved. Thus by assuming the fictitious domain characterised by 10, it is clear that the eventual MPr update equations should perform at least equal or better in the actual system since the bulk arriving channel releasing processes are not delayed.

#### 4.2.2 Update of $R_{RHS_i}$

With the overall channel releasing process characterised, we designed a simple scheduling algorithm to dispatch the released resources to one of reservation pools for call classes which require handoff support. Referred to as the Weighted Probability Dispatcher, it is described as follows:

*Weighted Probability Dispatcher:* For each unit of channel returning to the system, the dispatching mechanism chooses token buffer  $i$  with probability  $B_i \lambda_{hi} / \sum_i B_i \lambda_{hi}$  as the rightful token buffer.

If the rightful class  $i$  reservation pool is not full, the released channel is immediately reserved as a class  $i$  channel. Otherwise the channel is relegated as a common pool channel. By using this dispatching mechanism, it is clear that the *channel refilling rate* of the reservation pool for class  $i$  handoffs is as follows:

$$\mu'_{ri} = \mu'_r \times \frac{B_i \lambda_{hi}}{\sum_i B_i \lambda_{hi}} \quad (11)$$

where  $\mu'_r$ , as defined in 10, is the overall rate of the channel releasing process associated with the fictitious system.

For class  $i$  handoff calls with bandwidth requirement  $B_i$ , the *token refilling rate* is  $\mu'_{ri}/B_i$ , by counting  $B_i$  unit of bandwidth as one token.

Therefore, the appropriate reservation pool size to support class  $i$  handoff calls with QoS requirement in terms of worst handoff failure probability  $P_{QoS}$  is

$$R_{RPS_i} = B_i \times \left( \frac{\ln \frac{P_{QoS_i} + \rho'_i - 1}{P_{QoS_i}}}{\ln \rho'_i} - 1 \right) \quad (12)$$

where

$$\rho'_i = \frac{\lambda_{hi}}{\mu'_{ri}/B_i} = \frac{\lambda_{hi}B_i}{\mu'_{ri}} \quad (13)$$

In summary, by the inclusion of the Weighted Probability Dispatcher and the periodic updating of  $\lambda_{hi}$  and  $\lambda_{ai}$ , i.e. loading conditions, the required reservation pool sizes for the various class of services can be adaptively adjusted by virtue of 12. This sums up the MPr scheme for handoff support in a resource sharing Multi-Service environment. Finally, it is noted that in the MPr scheme of Figure 7, the reservation pools are clearly distinguished for the different class of services.

## 5 Numerical Results

The simulation model used here is a hexagonal system with three rings of radio cells as shown in Figure 9. We tested two scenarios, one for single service and the other with two call types. Both the call life time and channel holding time are set to be exponentially distributed. We tag individual cells and obtain performance metrics by simulation. Note the generic feature of the MPr scheme that there is no requirement that all cells must be MPr-compliant.

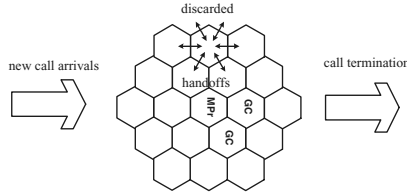


Figure 9: A cellular system with three rings of radio cells

### 5.1 A Homogeneous Call Scenario

Figure 10 shows the call dropping probability( $CDP$ ) and call blocking probability( $CBP$ ) of the MPr scheme with a target  $CDP$  below 0.05 for a homogenous call environment. Each call occupies 1 unit of bandwidth. The total capacity for a base station is 30. The mean call lifetime is 200s and the mean channel holding time for a call to stay in one cell is 100s.

For comparison, we present the results of the GC scheme with 2 and 3 guard channels, as they achieve comparable performance in terms of call dropping probability as well as call blocking probability, under light or mild load condition. When load increases, the  $CDP$  of GC scheme soars. However, the MPr scheme persists in keeping the  $CDP$  below the target level. By ensuring the handoff dropping QoS, a tradeoff occurs where the corresponding blocking probability of the MPr scenario is higher than either guard channel scenarios, because more channels are reserved for handoff calls. It has been proved

It should also be noted that for a stated QoS specification on call dropping, the MPr scheme is self-driven while the GC scheme requires either table lookup or two dimensional state-transition analysis as presented in [2, 3].

### 5.2 The Strict Dropping Policy

The performance of the strict dropping policy in a homogeneous call environment is also presented in Figure 11. In strict dropping policy, handoff calls are accepted only when there are channels available in the reservation pool, whereas in normal dropping policy, handoffs are accepted as long as there are unused channels. The strict dropping shows a worse case than the normal admission condition for handoff calls. It proves that the correct amount of resources is reserved to guarantee the QoS specification against handoff dropping by the MPr scheme. The lower call dropping probability for the normal admission under low load condition results from the competition for unused common channels, which are not guaranteed to be available under all loading conditions. Although it is shown in Figure 11a the expected QoS on call dropping is not violated, the strict dropping policy is not encouraged for the same reasons as discussed in Section 1.

### 5.3 A Multi-service Scenario

We tested a multi-service scenario with two call types. Both types of call may use all the available channels for the cell. The case of the complete partitioning environment is not presented as it is a trivial extension of the single-service scenario. The specifications of the two call types are shown in Table 1. Note that the wide-band call arrival rate is four times smaller than a narrow-band call and its handoff QoS requirements more stringent than the narrow-band call. Also note that the two types of calls have different motion characteristics.

Call Parameters	Type I	Type II
mean call life time	300s	300s
mean dwelling time	100s	150s
bandwidth requirement	1	12
call arrival rate	$\lambda_1$	$\lambda_2 = 0.25\lambda_1$
target maximum call dropping	0.10	0.05

Table 1: Parameters for two types of calls

Figure 12 illustrates the call dropping probabilities obtained using the MPr scheme in comparison with GC schemes. We notice the overwhelming favouring of narrow-band calls compared to wide-band calls in the GC cells. However, the MPr compliant cells do not over-favour the narrow-band calls, but maintain the stated QoS against handoff dropping for each type of calls, i.e. 0.10 for narrow-band calls in Figure 12(a) and 0.05 for wide-band calls for all loading conditions Figure 12(b).

## 6 Conclusion

In this paper we presented a novel measurement-based pre-reservation scheme, known as MPr, to reduce handoff failure in mobile multimedia networks with mixed-call types. It is capable of supporting multiple connection-level QoS for handoff calls of different types. In addition, the MPr scheme is very simple, efficient, and amenable for implementation in any heterogeneous cellular network where not all cells are MPr compliant.

## References

- [1] A.S. Acampora, and M. Naghshineh, "Control and Quality-of-Service provisioning in high speed microcellular networks", IEEE Personal Communications, vol. 1, no.2, 1994.
- [2] E. C. Posner and R. Guerin, "Traffic Policies in Cellular Radio that Minimize Blocking of Handoff Calls", ITC-11, Kyoto, 1985

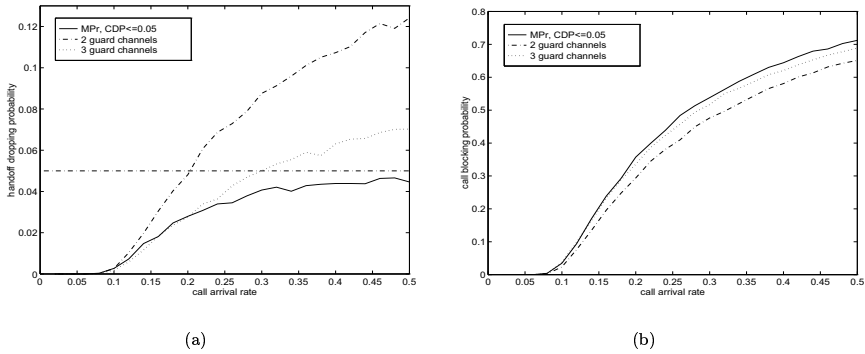


Figure 10: CDP and CBP v.s. arrival rate, single service

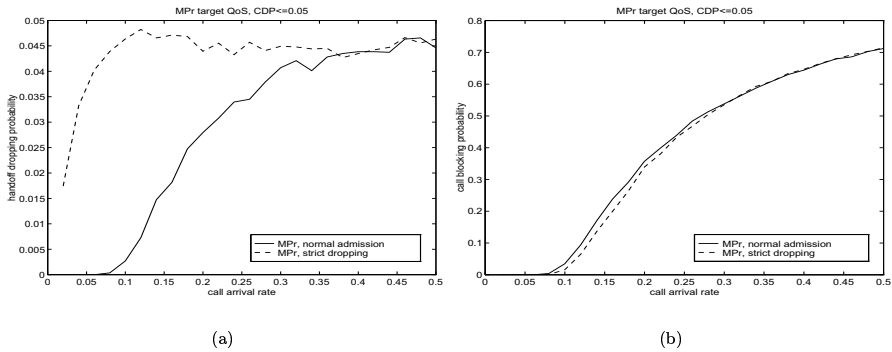


Figure 11: The strict dropping policy and normal admission for handoff

- [3] D. Hong and S. S. Rappaport, "Traffic Model and Performance Analysis for cellular mobile radio telephone systems with prioritized and non-prioritized handoff procedures", IEEE Transactions on Vehicular Technology, vol. 35, 1986, no. 3, pp. 77-92.
- [4] Sirin Tekinay, Bijan Jabbari "A Measurement-Based Prioritization Scheme for Handovers in Mobile Cellular Networks", IEEE JSAC, vol 10, No. 8, October 1992.
- [5] Stephen S. Rappaport and C. Purzynski, "Prioritised resource assignment for mobile cellular communication systems with mixed services and platform types", IEEE Trans. Veh. Tech., Vol.45, No.3, 1996.
- [6] B. Epstein and M. Schwartz, "Reservation Strategies for Multi-Media Traffic in a Wireless Environment", IEEE Vehicular Technology Conference 1995, pp.165-169, 1995.
- [7] J. Chen, M. Schwartz, "Reservation strategies for multi-media traffic in a wireless environment, performance summary", PIMRC'95, 1995
- [8] M. Naghshineh, A. S. Acampora, "QoS Provisioning in Micro-Cellular Supporting Multimedia Traffic", INFOCOM 1995

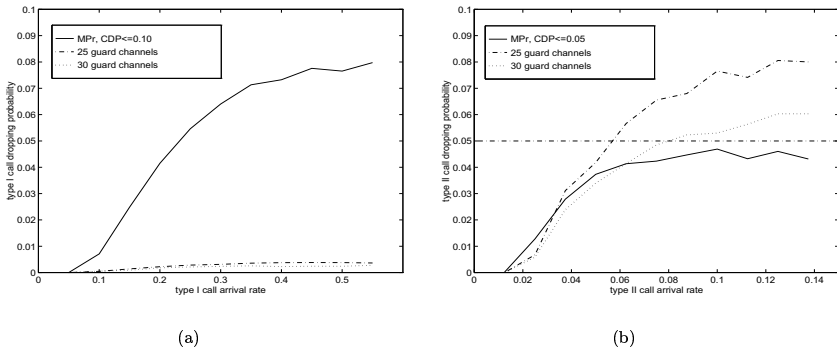


Figure 12: *CDP* v.s. arrival rate, two types of calls

- [9] A. Aljadhari, T. F. Znati, "A Predictive Bandwidth Allocation Scheme For Multimedia Wireless Networks".
- [10] Zhuang, W.; Chua, K.C.; Jiang, S.M., "Measurement-Based Dynamic Bandwidth Reservation Scheme for Handoff in Mobile Multimedia Networks", IEEE ICUPC '98. Vol 1 , pp311-315
- [11] M. Zonoozi, and P. Dassanayake, "User Mobility Modeling and Characterization of Mobility Patterns", IEEE JSAC, Vol. 15, No. 7, September 1997, pp. 1239-1252
- [12] D. A. Levine, I.F. Akyildiz and M. Nagshineh, "A Resource Estimation and Call Admission Algorithm for Wireless Multimedia Networks Using The Shadow Cluster Concept", IEEE/ACM Transaction on Networking, Vol. 5, No.1, February 1997, pp1-12.
- [13] J. Misis, S. T. Chanson, F. S. Lai, "Admission Control for Wireless Networks with Heterogeneous Traffic using Event Based Resource Estimation", ICCCN 1997. IEEE, Piscataway, NJ, USA, 97TB100187.. p 262-269
- [14] R. Ramjee, R. Nagarajan, and Don Towsley, *On Optimal Call Admission Control in Cellular Networks*, INFOCOM'96, pp43-50.
- [15] M. Nagshineh and M. Schwartz, "Distributed Call Admission Control in Mobile/Wireless Networks", IEEE JSAC, Vol 14, No. 4, May 1996, pp711-716
- [16] Chi-chao Chao, Wai Chen, "Connection admission control for mobile multiple-class personal communications networks", IEEE JSAC v15 n8 Oct 1997. p1618-1626.
- [17] C. Oliveira, J.B. Kim, T. Suda, "An Adaptive Bandwidth Reservation Scheme for High Speed Multimedia Wireless Networks", IEEE JSAC v16 n6 Aug 1998. p 858-874
- [18] Alberto Leon-Garcia, *Probability and Random Processes for Electrical Engineering*, 2nd Edition, Addison-Wesley.
- [19] Y.H. Kim, and C. K. Un, "Analysis of Bandwidth Allocation Strategies with Access Restriction in Broadband ISDN", IEEE Transactions on Communication, Vol. 41, No.5 May 1993, pp. 771-781

# QoS Rewards and Risks: A Multi-market Approach to Resource Allocation\*

Errin W. Fulp and Douglas S. Reeves

Departments of Computer Science & Electrical and Computer Engineering  
North Carolina State University, Raleigh NC 27695, USA

**Abstract.** A large number of network applications require a particular Quality of Service (QoS), that can be provided through proper network resource allocation. Furthermore, certain applications (multimedia oriented) may require guarantees of resource availability for predictable QoS. This paper introduces a distributed multi-market approach to network resource allocation. In this approach link bandwidth is bought and sold in two types of markets: the reservation market and the spot market. Together, these markets provide bandwidth guarantees and immediate availability. In addition, users have more flexibility when purchasing bandwidth that will maximize their individual QoS. Experimental results, using actual MPEG-compressed traffic, will also demonstrate the rewards and risks associated with purchasing various amounts in the reservation and spot markets.

## 1 Introduction

Many network applications require a certain Quality of Service (QoS) from the network for their proper operation. QoS may include bounds on packet delay, loss and jitter. The network can provide QoS by properly allocating its resources, such as link bandwidth, processor time, and buffer space. Furthermore, some applications may require a guarantee of resource availability for a duration of time. For example, the QoS of an application may be sensitive not only to the amount of link bandwidth allocated, but also any changes in the allocated amount that may occur. Due to the finite supply of resources and the various demands, fairly and efficiently allocating resources to provide QoS is a challenging problem.

Recently, microeconomics has been applied to network resource allocation and flow control. Congestion pricing is a well-known microeconomic approach that charges users for their consumption of resources, and prices are set based on supply and demand [1,4,6,11]. Alternatively, prices can be set with respect to

---

\* This work was supported by AFOSR (grants F49620-96-1-0061, F49620-97-1-0351 and F49620-99-1-0264) and DARPA Tolerant Networks Program of ITO. The views and conclusions contained herein are those of the authors and should not be interpreted as necessarily representing the official policies or endorsements, either expressed or implied, of the AFOSR, DARPA or the U.S. Government.



marginal costs [7]. With such a model, prices can be calculated in a distributed fashion to encourage high utilization of network resources as well as a fair distribution. However in many cases, the transient behavior and the method of distributing intermediate prices (and/or allocations) during convergence is generally ignored. Furthermore, many have not been validated in a detailed way using realistic network configurations and actual traffic.

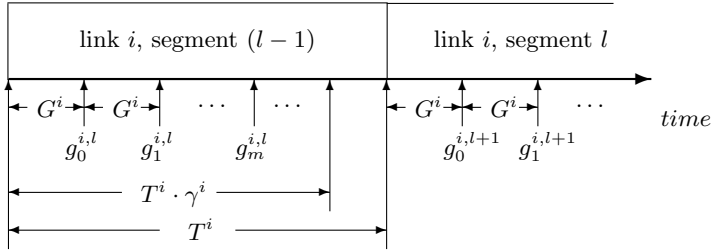
In [3] we introduced a congestion pricing technique based on the competitive market model. Unlike other congestion pricing methods, this distributed technique allows changing traffic demands by dynamically adjusting the link bandwidth price in response to current supply and demand. It has been proven and demonstrated that this method can achieve Pareto-optimal and fair allocations as well as high link utilization [2]. Although able to achieve optimal allocations under dynamic conditions, a limitation of this method (shared with other methods) is that no guarantee of future bandwidth availability can be made. This may be acceptable for “elastic-traffic” that can easily adjust demands based on network conditions [12]. For other applications (such as high definition video), changing bandwidth amounts may result in a sudden reduction of QoS or in the worst case, being forced out of the economy due to high prices. Therefore, it is advantageous to shield these users from unpredictable price fluctuations.

In this paper, a *multi-market* approach for bandwidth allocation is presented. Multi-market methods have been proposed for allocating link bandwidth and buffer space [5,11]; however the primary focus was not to provide price or QoS stability. In our approach link bandwidth is bought and sold in two types of markets: the reservation market and the spot market. In the reservation market, bandwidth is bought and sold in amounts for a duration of time. Bandwidth purchased in the reservation market may be used to provide some required (minimal) QoS. In the spot market, a user can immediately use whatever amount of bandwidth they find affordable, with no reservation overhead. Therefore, this multi-market approach combines the unique advantages of the single spot market [2] (such as immediate availability) with the price stability offered in the reservation market. This is done in a distributed fashion with a state-less implementation. The flexibility of the multi-market approach is well suited to accommodate the diverse QoS needs of various elastic and non-elastic network applications.

The remainder of this paper is structured as follows. Section 2 describes the new multi-market technique in detail. Section 3 provides a demonstration of multi-market allocation method. Finally, section 4 reviews the multi-market technique, summarizes the results and discusses some open questions.

## 2 A Multi-market Approach

Similar to the spot market economy [3], the multi-market economy is based on a competitive market model, where pricing is done to promote Pareto-optimal and fair distributions, as well as high utilization [8]. However, unlike the spot market approach the multi-market provides guarantees of resource availability



**Fig. 1.** Example segments and price calculation points for link  $i$  reserved bandwidth.

as well as price and QoS stability. This results in an allocation method that is appropriate for elastic and non-elastic traffic.

There are three entities in this network economy: users (those who execute network applications), Network Brokers (NB) and switches. Using the competitive market nomenclature, users are consumers, switches are producers and network brokers are used to assist the exchange of resources in the market. While there are many resources in a computer network, this paper focuses on the pricing of link bandwidth.

## 2.1 Switch

The network consists of several switches interconnected with links. For a unidirectional link between two switches, we consider the sending switch as owner of the bandwidth of that link. For link  $i$  denote the total capacity of link bandwidth as  $S^i$ . The capacity is then divided into two types, *reserved* and *spot* bandwidth. Reserved bandwidth is sold as an amount for a duration of time, while spot bandwidth is sold as a non-storable resource. With this distinction, reserved and spot bandwidth are considered separate resources. Reserved bandwidth has the unique advantage of ownership over a period of time, while the advantage of spot bandwidth is its immediate availability. Each resource is sold in its own local (link) market, therefore the switch will associate two markets per output port (thus the “multi-market” designation). These markets operate independently and asynchronously since there is no need for market communication (for example, price comparisons) or synchronization from switch to switch. Since the link capacity is divided into reserved and spot bandwidth, the switch must differentiate the traffic using either type. We assume that a bit will be set in the header of the packet, indicating the packet is using reserved bandwidth.

**Reserved Bandwidth Market** Link  $i$  will sell a maximum fraction  $\beta^i$  of  $S^i$  as reserved bandwidth. The reserved bandwidth is divided into equal non-overlapping intervals of time called segments, where the length of each segment is denoted as  $T^i$ . Portions of the segment are then sold to users with an auction procedure. Users are only able to bid for an amount of the next segment; therefore, the reserved bandwidth of segment  $l$  is auctioned during segment  $(l-1)$ . At

the beginning of the auction for segment  $l$  of link  $i$ , users forward a bid (for an amount of reserved bandwidth they wish to purchase) to the switch. The sum of these bids, denoted as  $h_m^{i,l}$ , is recorded by the switch and is used to update the price. During the auction, the price of reserved bandwidth for segment  $l$  is adjusted at regular intervals  $G^i$ , as seen in figure 1. The  $m$ th auction price for reserved bandwidth of link  $i$ , segment  $l$  is denoted as  $g_m^{i,l}$ . The price for the next interval is calculated using the following tâtonnement process,

$$g_{m+1}^{i,l} = g_m^{i,l} \cdot \frac{h_m^{i,l}}{\beta^i \cdot S^i} \quad (1)$$

The tâtonnement process adjusts the price at regular intervals, based on the demand ( $h_m^{i,l}$ ) and the supply ( $\beta^i \cdot S^i$ ). If the demand is greater than the supply, then the price increases (and vice versa). After a new auction price is calculated, it is distributed to NB's, who may submit updated bids. It is important to note the switch does not need to store individual bids. Users can initially submit a bid amount, then send only changes (differences) to the switch. This process repeats until an *equilibrium price*  $g_*^{i,l}$  for segment  $l$  is determined. An equilibrium price causes demand to equal the supply. At this price bandwidth is sold and the resulting allocation is Pareto-optimal and fair [2]. New bidders (users who have not yet participated in bidding for the segment) are not allowed to participate after  $T^i \cdot \gamma$  (where  $0 < \gamma < 1$ ) has passed. This provides time for the auction to converge to the equilibrium price before the segment begins. At the end of auction the switch notifies the users that a new segment has begun<sup>1</sup>. Users are then able to use the amount of reserved bandwidth they defined in their last bid (explicit notification is not necessary). Since only aggregate information (not individual) is used and it is not necessary for the switch to store individual bids, the auction process for the reservation market can be considered a state-less implementation.

**Bandwidth Spot Market** As described in [3], bandwidth in the spot market is considered a non-storable resource (similar to electricity); therefore users/NB's are unable to purchase spot bandwidth in hopes of using it a later time. Once a NB has determined an affordable amount of spot bandwidth, the user can send immediately (no reservation overhead is required)<sup>2</sup>.

The spot market price for link  $i$  is calculated at the switch, at discrete intervals of time. The price during the  $n$ th interval is constant and is denoted as  $p_n^i$ . At the end of the  $n$ th spot market price interval, denote the demand for spot market bandwidth as  $d_n^i$  and the amount of reserved bandwidth currently used as  $r_n^i$  where  $r_n^i \leq \beta^i \cdot S^i$ . The total spot market supply for link  $i$  is  $S^i - r_n^i$ ; therefore any reserved bandwidth that is not used can be sold in the spot market. At the

<sup>1</sup> Methods of price distribution and user notification, as well as queueing delay issues are discussed in [2].

<sup>2</sup> Without reservations, selling the same spot bandwidth to multiple users may occur. How this is avoided is described in [2].

end of the  $n$ th interval, the switch updates the spot market price of link  $i$  using a *modified tâtonnement process* [3]. A limitation of the tâtonnement process, in its original form (equation 1), is the inability to dynamically adapt to changing demands<sup>3</sup>. To handle such dynamics, the spot market price is determined using the following modified tâtonnement process.

$$p_{n+1}^i = p_n^i \cdot \frac{d_n^i}{\alpha^i \cdot S^i - r_n^i} \quad (2)$$

The modified tâtonnement process adjusts the price at regular intervals, based on the demand (current spot traffic) and the supply. The bandwidth supply is the total bandwidth times a constant  $\alpha^i$  (where  $0 < \alpha^i \leq 1$ ) minus the amount of reserved bandwidth currently used ( $r_n^i$ ). This modification causes the price to increase after some percentage ( $\alpha^i$ ) of the available spot bandwidth has been sold. This is evident from the equation, since the price will only increase if the numerator is greater than the denominator ( $d_n^i > \alpha^i \cdot S^i - r_n^i$ ). Once the new price  $p_{n+1}^i$  is calculated it is distributed to the users/NB's using the link. Upon receiving the new price users/NB's adjust their transmission rate. As demand changes, the modified tâtonnement process dynamically adjusts the price seeking the new equilibrium price  $p_*^i$ . Similar to the reservation market, the spot market is state-less since the bandwidth price is calculated using only the local aggregate demand, supply and current price.

## 2.2 User

A user, executing a network application, requires link bandwidth for transmission. The amount of bandwidth desired (or the desired rate) is determined from the application and is denoted as  $a$ . Based on prices and wealth, the user can afford a range of bandwidth (less than or equal to  $a$ ), and some amounts will be preferred over others. In economics these preferences are represented with a utility function. The utility function maps a resource amount to a real number, that corresponds to a satisfaction level. The utility curve can be used to compare resource amounts based on the satisfaction the user will receive. For this economy we will use *QoS profiles* [9] for the utility curves. The profile can be approximated by a piece-wise linear curve with three different slopes, as seen in figure 3(b). The horizontal axis measures the bandwidth ratio of allocated bandwidth to desired bandwidth. The vertical axis measures the satisfaction and is referred to as a QoS score. Our QoS scores range from one to five, with five representing an excellent perceived quality and one representing very poor quality.

Since there are two different types of resources in the economy (reserved and spot bandwidth), the user must identify how the reserved and spot bandwidth may be substituted for one another. In microeconomics these preferences

<sup>3</sup> In the reservation market the equilibrium price must be determined before bandwidth is sold; therefore, if demands change a new equilibrium price must be determined.

are represented with an indifference curve. An indifference curve indicates the combinations of resources that result in the same utility [8]. For our economy, indifference curves indicate the combination of reserved and spot bandwidth that result in the same utility. These curves are normalized to the current desired amount of bandwidth  $a$ , as seen in figure 2(a). For example in figure 2(a), the indifference curve labeled “prefer-reserved” was generated from the equation  $y = k \cdot (1 - x)^2$ , where  $y$  is the amount of spot bandwidth and  $x$  is the amount of reserved bandwidth. The preference for reserved bandwidth increases as  $k$  increases. The other indifference curve given in figure 2(a) represents a user who has no preference for reserved or spot bandwidth. In this case, spot and reserved bandwidth are considered “perfect substitutes” and the user will always prefer the *cheaper* (lower cost) bandwidth. In the case where the two types of bandwidth have the same price, we assume the user will prefer equal amounts of either type of bandwidth. The only assumptions required for the indifference curve is that it must be continuously differentiable and convex to the origin (required for determining the amount of bandwidth to purchase).

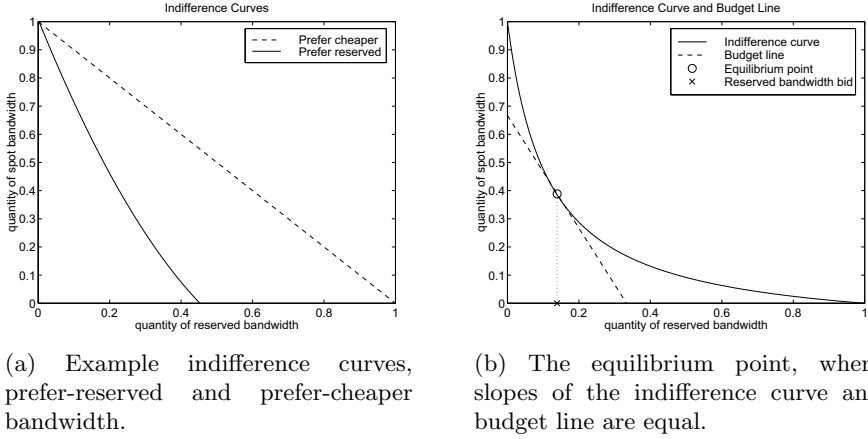
Finally, the user is charged continuously for the duration of the session (analogous to a meter). To pay for the expenses, we will assume the user provides an equal amount of money over regular periods of time [2]. We will refer to this as the budget rate of the user,  $W$  (\$/sec).

### 2.3 Network Broker

Users can only participate in the network economy through a network broker (NB). This entity is an agent for the user and is located between the user and the edge of the network. Representing the user in the economy the NB performs the following tasks: connection admission control, policing, packet marking, and purchasing/bidding decisions. Although the NB works as an agent for the user (making purchasing decisions), we assume that the NB operates honestly in regards to both the switches and the user.

The NB monitors the user and the prices by gathering and storing information about each. From the user, the NB collects and stores; the QoS profile, indifference curve,  $a$  and  $W$ . The NB also stores the route,  $R$ , that connects source to the destination, where  $R$  consists of  $v$  links,  $\{l^i, i = 1 \dots v\}$ . For each link on  $R$ , the NB collects current reserved and spot bandwidth prices<sup>4</sup>. The NB will divide the budget rate,  $W$ , into a vector of  $v$  budget rates  $\mathbf{w}$ , where  $\mathbf{w} = \{w^i, i = 1 \dots v\}$  and  $w^i$  corresponds to link  $i$ . Separate budgets are used to localize the effect of prices to each link. Using this information the NB controls network admission by initially requiring the user to have enough wealth to afford at least an *acceptable* QoS; otherwise, the user is denied access. The NB also levies the user for their consumption. In addition, the NB polices the user, ensuring only the bandwidth purchased is used, and marks the packets (assigning which are to use reserved bandwidth). Finally, the NB determines the reserved bandwidth bid and the amount of spot bandwidth to purchase.

<sup>4</sup> The requirement that the NB must know the entire route, and store a distinct price per link, can be relaxed [2].



**Fig. 2.** Example indifference curves and how they are used.

**Bidding and Purchasing Bandwidth** Since reserved bandwidth is sold in an auction format, the NB must bid for reserved bandwidth at each link on the route. The bid (the amount the user will purchase) for link  $i$  is based on the budget  $w^i$ , a statistic of the spot bandwidth price, and the reserved bandwidth auction price for this link  $g^{i,l}$ . A statistic is required for the spot bandwidth price due to its volatility. For our discussion (and simulation) we will use the maximum spot price,  $\hat{p}^i$ , measured during the current segment. Using this information the NB maximizes the utility of the user  $u(x, y)$ ,

$$\max \{u(x, y)\}, \quad g^{i,l} \cdot x + \hat{p}^i \cdot y \leq w^i \quad (3)$$

where  $x$  is the amount of reserved bandwidth and  $y$  is the amount of spot bandwidth. As defined in [8], the first order condition of this constrained maximization problem is,

$$\frac{\partial u / \partial x}{\partial u / \partial y} = \frac{g^{i,l}}{\hat{p}^i} \quad (4)$$

The NB must spend the budget to equalize the ratio of marginal utility to the price of each resource. Plotting the budget line with the indifference curve, as seen in figure 2(b), the slope of the budget line is  $g^{i,l} / \hat{p}^i$ . Therefore, the point where the slope of the indifference curve and the budget line are equal is where the utility of the user is maximized. The second order condition is not required due to the convexity assumption of the indifference curve [8]. The reserved bandwidth bid is the  $x$  component of this point and is forwarded to link  $i$ .

The NB keeps a table storing the amount and price of reserved bandwidth purchased at each link in the route. When the segment for link  $i$  is sold, the amount purchased  $c^i$  and the price  $f^i$  is updated in the table. The maximum

amount of reserved bandwidth that can be used is,

$$e = \min_{i=1\dots v} \{c^i\} \quad (5)$$

which is the minimum amount of reserved bandwidth purchased at any link. If the desired bandwidth is greater than the purchased reserved bandwidth, then spot bandwidth is used for transmitting the remaining portion  $(a - e)$ . The amount of spot bandwidth to use  $y$  is,

$$y = \min \left\{ \min_{i=1\dots v} \left\{ \frac{w^i - e \cdot f^i}{p_n^i} \right\}, a - e \right\} \quad (6)$$

which is the maximum amount of spot bandwidth that is affordable, but no more than what is required  $(a - e)$ .

## 2.4 Optimality and Network Dynamics

For any allocation method it is important to address the optimality and fairness that is achievable. For an economy consisting of multiple competitive markets, once the tâtonnement processes (one per market) reach equilibrium, the resulting allocation is Pareto-optimal and fair [2]. Due to the nature of multimedia traffic, it is also important to determine if the allocation technique can adequately handle network dynamics (users entering/exiting and variable bit rate sources). Due to the complexity of actual network dynamics, simulations have been used to demonstrate the performance of the competitive market approach. Experimental results have shown the spot market achieves optimal allocations over 90% of the time under realistic network conditions [3]. In the next section, simulation is used to demonstrate the performance of the multi-market economy under similar conditions.

## 3 A Demonstration of the Multi-market Economy

In this section the performance of the multi-market network economy is demonstrated via simulation. Experiments performed will consist of a realistic network configuration, allowing users to randomly enter the network and use actual MPEG-compressed traffic. Simulation results will show that users who prefer reservations will experience less of a QoS impact than those who do not. Results also show that the preference for reservations is at the expense of lower average QoS. This represents the *QoS rewards and risks* of the multi-market.

The network simulated consisted of 160 users and their associated NB's, four switches and seven primary links, as seen in figure 3(a). Each output port carried traffic from 40 users and connected to a 45 Mbps link. Links interconnecting switches were 1000 km in length, while links connecting sources to their first switch were 25 km in length. Users had routes consisting of one, two or three hops. The network can be described as a "parking lot" configuration, where

multiple sources use one primary path. The multi-market economy had the following initial values. The spot market parameter  $\alpha$  (targeted utilization) was 90%. Switches sold equal amounts of reserved and spot bandwidth; therefore  $\beta$  was 45%. Reserved bandwidth prices were initialized to 55 and segments were 15 minutes in duration. Longer segments could have been selected; however, we were interested in observing the transition effects from one segment to another and the smaller segment size reduced the simulation time. Spot market prices were initialized to 50 and the update interval was 20 times the longest propagation delay. Assume no propagation delay between the user and their NB.

Users had budget rates<sup>5</sup>,  $w$ , of  $3 \times 10^8$ /sec and used the QoS profile given in figure 3(b). Since all users have the same budget rate, they are considered equal (purchasing power) in the economy [2]. The source for each user was one of 15 MPEG-compressed traces obtained from Oliver Rose at the University of Würzburg, Germany [10]<sup>6</sup>. Although users have the same wealth and QoS profile, for this demonstration users are considered either *long-term* or *short-term*. Long-term users measure, over the duration of the simulation, the different QoS obtained from preferring different amounts of reserved and spot bandwidth. Short-term users are introduced in the economy to cause sudden demand shifts, which may occur in actual networks (peak load times). Together, we are interested in the QoS achieved by the various users in the multi-market economy.

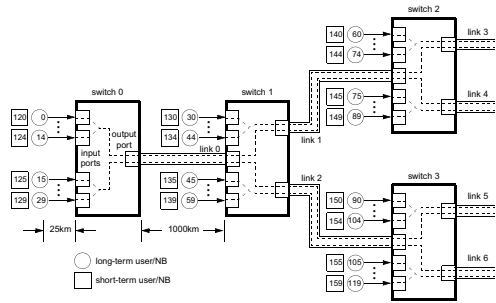
A total of 120 users are considered long-term and have sessions that last the duration of the simulation. Half of the long-term users prefer reserved bandwidth, the remaining prefer cheaper bandwidth (indifference curves given in figure 2(a)). The long-term users enter the network at random times uniformly distributed between 0 and 600 seconds. The remaining 40 users are considered short-term. These users transmit a short segment of an MPEG video (under 3 minutes, randomly determined). Due to the relative shortness of their session, these users will only purchase bandwidth from the spot market. Starting at 3000 seconds the short-term users enter the network with a Poisson distribution of mean 120 seconds. We are interested in the link bandwidth utilization and the QoS provided to each type of long-term user. Allocation graphs are provided to measure the utilization of link bandwidth, while QoS graphs measure the average QoS observed by long-term users.

For this simulation, the example bandwidth allocations, prices and average QoS are given in figures 3(c) - 3(e). The results from 1800 to 6500 seconds are displayed since we are only interested in the effect the short-term users have in the economy. As seen in figure 3(c), all of the available reserved bandwidth for link 3 was sold, while the total bandwidth allocated stayed within the vicinity of  $\alpha$  (targeted utilization). Similar results were noted for the remaining links. As seen in figure 3(e), before the short-term users entered the network (time less than 3000 seconds) prefer-cheaper users enjoyed a higher QoS. During this time, prefer-cheaper users only purchased bandwidth from the spot markets,

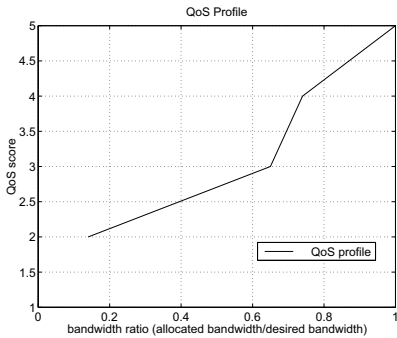
<sup>5</sup> The denomination is based on bps, if based on Mbps the budget would be 300/sec.

<sup>6</sup> Traces can be obtained from the ftp site <ftp-info3.informatik.uni-wuerzburg.de> in the directory `/pub/MPEG`

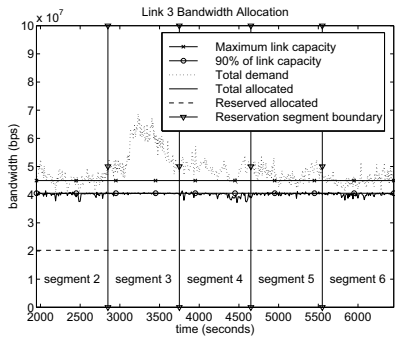




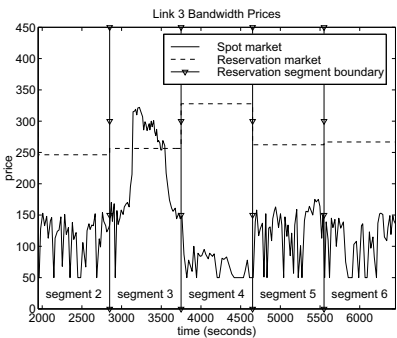
(a) Network configuration used for simulations.



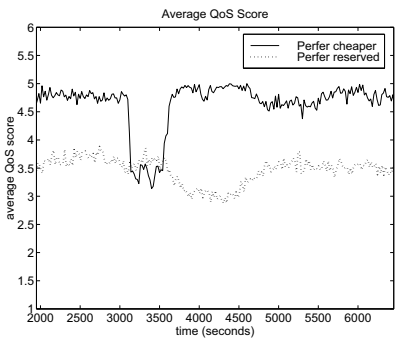
(b) QoS profile.



(c) Link 3 allocation.



(d) Link 3 bandwidth prices.



(e) Average QoS scores for all long-term users.

**Fig. 3.** Multi-market simulation set-up and results.

since prices were lower (as seen in figure 3(d)). In contrast, prefer-reserved users spent their entire budget in the reservation markets. Purchasing from the spot markets yielded higher QoS for the prefer-cheaper users, because these users only purchased what they needed at any time. This allows users to efficiently share the spot market bandwidth. In the case where the total demand of the prefer-cheaper users exceeded the spot market supply, each prefer-cheaper user received an equal-share of the spot market bandwidth. The prefer-reserved users were allocated an equal-share of the reserved bandwidth supply for the duration of the simulation (no more). For this reason, prefer-reserved users observed a lower QoS, until the short-term users arrived. When the short-term users arrived (during segment 3 of figure 3(c)), the spot market price increased in response to the increase in demand. During this time prefer-cheaper users received a lower QoS, since they had to compete with the new arrivals. The standard deviation in QoS observed by the prefer-cheaper users during this segment was 0.65, which was six times higher than the prefer-reserved users. The prefer-reserved users continued to receive approximately the same level of QoS since they purchased reserved bandwidth for segment 3. The spot market price increase, during segment 3, did cause a higher reservation market price for segment 4, as seen in figure 3(d). The prefer-reserved users could afford less reserved bandwidth during this segment, resulting in a slightly lower QoS. Afterwards, prices and QoS observed by the users returns to the previous values.

This simulation provides some insight to the rewards and risks of purchasing various amounts of bandwidth in the spot and reservation markets. In the example, prefer-cheaper users enjoyed the reward of a higher QoS (average of 4.65), but were susceptible to the risk of spot market price fluctuations. Applications that can gracefully handle such unpredictable bandwidth changes (such as teleconferencing) would perform better by accepting the risk associated with the spot market (which was typically cheaper). In contrast, prefer-reserved users opted for the more stable reservation market, but generally received a lower QoS (average of 3.46). Applications, such as high definition video, that can not adapt well to sudden bandwidth fluctuations would perform better with the stability provided from the reservation market.

## 4 Conclusions

This paper introduced a decentralized bandwidth allocation method based on a multi-market economy. A computer network can be viewed as an economy consisting of three entities (users, Network Brokers and switches) and two different markets/resources (reserved and spot bandwidth). Switches own the bandwidth sought by users, which is sold in the reservation and spot markets. Reserved bandwidth has the advantage of ownership over a period of time, providing the user with some predictability of their expected QoS. In contrast, spot bandwidth has the advantage of immediate availability without the reservation overhead. Therefore, the multi-market approach integrates the benefits of the spot market and the reservation market in one allocation technique. Both market types

are modeled as competitive markets; therefore, Pareto-optimal and fair allocations are possible. The flexibility of the multi-market approach is well suited to accommodate the diverse QoS needs of various network applications (from elastic-traffic to applications that prefer predictability). Experimental results showed that users who preferred reservations experienced a more stable QoS than those who do not. However, the preference for reservations came at the expense of lower average QoS. Therefore this represents the QoS rewards and risks of the multi-market approach. Some areas for future work include the association of indifference curves with QoS profiles and dynamically adjusting market parameters in response to market trends.

**Acknowledgements.** The authors wish to thank Maximilian Ott and Daniel Reininger of C & C Research Laboratories, NEC USA for their significant contributions to this research.

## References

1. D. F. Ferguson, C. Nikolaou, J. Sairamesh, and Y. Yemini. Economic Models for Allocating Resources in Computer Systems. In S. Clearwater, editor, *Market Based Control of Distributed Systems*. World Scientific Press, 1996.
2. E. W. Fulp. *Resource Allocation and Pricing for QoS Management in Computer Networks*. PhD thesis, North Carolina State University, 1999.
3. E. W. Fulp, M. Ott, D. Reininger, and D. S. Reeves. Paying for QoS: An Optimal Distributed Algorithm for Pricing Network Resources. In *Proceedings of the IEEE Sixth International Workshop on Quality of Service*, pages 75 – 84, 1998.
4. F. Kelly, A. K. Maulloo, and D. K. H. Tan. Rate Control for Communication Networks: Shadow Prices, Proportional Fairness and Stability. *Journal of the Operational Research Society*, 49:237 – 252, 1998.
5. S. H. Low. Equilibrium Allocation of Variable Resources for Elastic Traffics. In *Proceedings of the IEEE INFOCOM*, pages 858 – 864, 1998.
6. J. K. MacKie-Mason and H. R. Varian. Pricing Congestible Network Resources. *IEEE Journal on Selected Areas in Communications*, 13(7):1141 – 1149, Sept 1995.
7. J. Murphy, L. Murphy, and E. C. Posner. Distributed Pricing for ATM Networks. *ITC-14*, pages 1053 – 1063, 1994.
8. W. Nicholson. *Microeconomic Theory, Basic Principles and Extensions*. The Dryden Press, 1989.
9. D. Reininger and R. Izmailov. Soft Quality-of-Service for VBR+ Video. In *Proceedings of the International Workshop on Audio-Visual Services over Packet Networks, AVSPN'97*, Sept. 1997.
10. O. Rose. Statistical Properties of MPEG Video Traffic and Their Impact on Traffic Modeling in ATM Systems. Technical Report 101, University of Würzburg Institute of Computer Science, Feb. 1995.
11. J. Sairamesh, D. F. Ferguson, and Y. Yemini. An Approach to Pricing, Optimal Allocation and Quality of Service Provisioning in High-speed Packet Networks. In *Proceedings of the IEEE INFOCOM*, pages 1111 – 1119, 1995.
12. S. Shenker. Fundamental Design Issues for the Future Internet. *IEEE Journal on Selected Areas in Communications*, 13(7):1176 – 1188, 1995.

# Spare Capacity Planning for Survivable Mesh Networks

Adel Al-Rumaih<sup>1</sup>, David Tipper<sup>1\*</sup>, Yu Liu<sup>1\*</sup>, and Bryan A. Norman<sup>2</sup>

<sup>1</sup> University of Pittsburgh, Department of Information Science and Telecommunications,  
Pittsburgh, PA 15260, USA

{alrumaih, tipper, yuliu}@tele.pitt.edu

<sup>2</sup> University of Pittsburgh, Department of Industrial Engineering, Pittsburgh, PA 15261  
banorman@engrng.pitt.edu

**Abstract.** The design of survivable mesh based STM networks has received considerable attention in recent years and is a complex multi-constraint optimization problem. In this paper, a new spare capacity planning methodology is proposed utilizing genetic algorithms. The method is based on forcing flows/traffic which are on paths that are disjoint to share backup spare capacity. The major advantages of the new approach are a polynomial time complexity and the capability of incorporating nonlinear variables such as nonlinear cost functions into the solution algorithm. Numerical results illustrating the form of the genetic algorithm solution and comparing the proposed methodology to existing techniques from the literature are presented.

## 1 Introduction

Due to the widespread use of telecommunications networks and society's growing dependence upon the exchange of information, the need for reliable communication service has become increasingly important. Several highly publicized outages have illustrated that disruption of communication services is very costly to businesses, governments and the general public. This has led to growing interest in the design of networks which are survivable [1-11].

One component in constructing a survivable network is designing the physical network with enough spare capacity to enable fault recovery via rerouting of traffic in the event of a failure. The problem of spare capacity placement for survivable mesh networks has recently been studied for STM, WDM, and ATM networks. In this paper, we consider the problem of given a STM mesh type network topology, the normal traffic demand, and the capacity allocation to meet the normal traffic demand, how much spare capacity should be provisioned and where should it be located in order for the network to tolerate a specified set of failure scenarios (e.g., loss of any single link). The term "mesh" doesn't imply that the network topology is a full mesh, but rather that the network nodes are at least two connected. Operational backbone networks typically have an average nodal degree in the range of 2.5 to 4.5 [5].

Previous research on spare capacity assignment in STM mesh networks adopts the problem context above and uses either mathematical programming techniques [1-7,

---

\* Supported in part by DARPA under No. F30602-97-1-0257 and NSF grant NCR-9506652

11] or heuristics [8-11] to determine the spare capacity allocation. In both cases, algorithms have been developed for link restoration and path restoration. In link restoration, the nodes adjacent to a failed link are responsible for rerouting the affected traffic flows around the failed link. Thus, one merely patches the hole in the original route. In contrast, for path restoration, the source destination node pairs whose traffic traversed the failed device are responsible for restoration and reroute over the entire path set between each affected source destination pair. In general, path restoration is known to require less spare capacity than link restoration [1,3-6]. However, path restoration is more complex to implement as many more nodes are involved in the restoration process. Also, in the general case path restoration requires the release of stub capacity (i.e., the capacity on the unaffected portion of a failed working path) both upstream and downstream of the failed device. A variation on path restoration which speeds up restoration and does not require stub release is *path restoration with link disjoint routes* [11]. In path restoration with link disjoint routes, no portion of the failed working path is used for backup routes, thus the restoration process can begin quickly without additional signaling overhead specifying exactly which component failed in the working path and confirming the release of stub capacity. This is an important advantage in current STM and WDM networks where fault identification signaling capabilities are limited.

In both link and path restoration one is interested in determining a set of backup routes for the working traffic and the amount of spare capacity required on the backup routes to achieve a desired level of traffic restoration for a specific failure scenario. A typical goal is to determine the spare capacity placement such that all normal traffic can be restored for any single link failure in the network. Mathematical programming methods have been used to formulate the spare capacity planning problem for link and path restoration approaches as Integer Programming (IP) models [2-6,11]. The objective function adopted is to minimize the spare capacity required for achieving restoration from a specific failure condition. Unfortunately, the resulting IP formulation is NP-hard and in order to solve the model sub-optimal heuristic approaches have been tried. One approach is to approximate the IP model by a Linear Programming (LP) model which has a polynomial time bound solution and then round the solution to integer values. An alternate approach is to solve the IP model exactly over a reduced search space. Typically, the search space is reduced by limiting the set of paths over which the backup routes are determined. For example, placing a hop count limit on the path set equal to the diameter of the network or a limit on the number of hops in addition to the shortest path hop count for each source-destination pair. How to pick the path set within a general setting in order to achieve good results while maintaining a computational feasible problem that scales is still an open problem. An additional limitation of the IP approach is the inability to incorporate nonlinear variables such as a nonlinear capacity cost function or quality of service (QoS) variables. In contrast to the mathematical programming approach, heuristics in the literature [5,8-11] essentially seek to quickly find a reasonable spare capacity assignment that meets the fault tolerance requirements.

Here we present a new spare capacity planning technique based on path restoration with link disjoint routes that uses genetic algorithms in the solution process. Genetic algorithms (GA) have received considerable attention in recent years for use in solving various optimization problems [12], including the solution of integer programming problems [13] and data network topology design [14, 15]. One

advantage of the approach is the ability to incorporate nonlinear variables in the solution process, such as realistic nonlinear capacity cost values. Additionally, the computational time is easily controlled allowing the approach to scale to large networks. The description of the proposed GA is given in Section 2. Section 3 presents a study of numerical results illustrating the application of the proposed GA technique, guidelines for parameter selection and computational complexity. Additionally, for the sake of comparison, numerical results are given for the standard IP approaches and a popular heuristic. Section 4 gives our conclusions.

## 2 A New Approach to Spare Capacity Design

Our methodology consists of using a GA to implement the concept that traffic flows which travel over disjoint routes may be able to share spare capacity on a backup path, since it is unlikely that more than one failure will occur simultaneously. Thus, our approach tries to reduce the cost of spare capacity needed for a specific fault tolerance requirement (e.g., full recovery from any single link failure) by finding a set of backup paths that enables the sharing of spare capacity. This can be achieved by forcing the backup paths to use a subset of the links in the network, thus concentrating the backup paths on certain routes of the network, which results in reducing the total cost due to the nonlinear economy of scale of capacity cost [5,17].

GAs are stochastic search techniques that mimic the survival of the fittest (or best) paradigm observed in nature [16]. A GA optimizes a fitness function (i.e., objective function) by evolving successive generations of a population utilizing breeding and mutation transformations to move from one generation to the next in a manner such that only the fittest (best solutions) survive from generation to generation. A GA first creates an initial population of solutions (i.e., a generation) by either using random solutions or solutions constructed using some heuristic procedure and these solutions represent the basis for the evolutionary improvement process. In our GA, selection of a new population from the current population is accomplished, as shown in Figure 1, by performing three operations: (1) elitist reproduction, (2) crossover, and (3) mutation. In elitist reproduction a portion of the current population is copied directly into the next generation to maintain progress of the search through successive generations. In crossover two members of the current generation are selected for breeding and are bred via a crossover operation creating a pair of offspring to add to the population. In mutation a portion of the population is randomly selected and stochastically altered in order to maintain diversity of the genetic search. The new population is evaluated (sorted according to the fitness function) and the new generation is selected from the best members of the population with the remainder being eliminated. This process is repeated until a stopping criterion is met, usually either a finite number of generations of the population or when the improvement in the fitness function between successive generations is less than a specified value.

There are many variations of GAs based on the choice of reproduction, crossover and mutation operators. Here we develop a GA for our problem domain based in part on the approach given in [13] for the solution of nonlinear IP problems. Before applying the GA to determine spare capacity, the paths to be used and the required capacity under normal conditions are determined based on the traffic demands, topology and shortest path routing [15, 19]. We use a two-stage algorithm to

implement the GA method for spare capacity planning. Stage I determines an initial population for the GA, in effect finding a set of feasible solutions to the spare capacity assignment problem. Stage II then tries to improve the spare capacity assignment resulting from Stage I via a GA search as shown in Figure 1. The two stages of the GA are briefly described below; detailed information is given in [18].

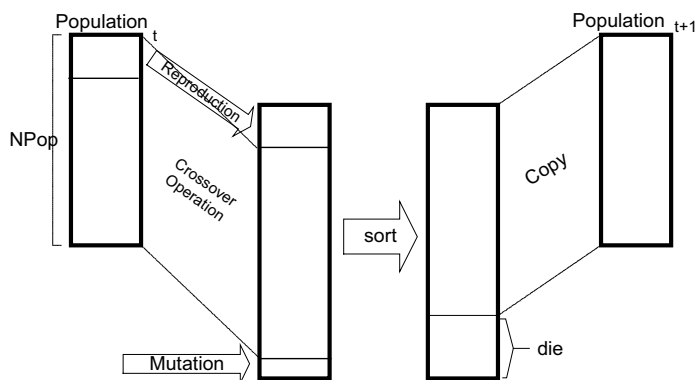


Fig. 1. Transition between two successive generations in a GA.

Consider a network of  $L$  links and  $N$  nodes. We represent the network topology by an adjacency matrix  $Top$  where element  $Top_{ij}$  is 1 if a link exists between node  $i$  and node  $j$ . The routes used to carry traffic under normal conditions are denoted by  $WP$  (working paths), where  $WP_{ij}$  is the path used between node  $i$  and node  $j$ . Similarly, we define  $BTop$  as the adjacency matrix that represents a virtual network topology used to calculate a set of backup paths  $BP$ . Let  $SA$  denote a set of spare capacity assignments that meet a required restoration level for a specific failure scenario. We define  $Npop$  as the size of the initial population and subsequent generations. Let  $SA^k$  denote the  $k$ th spare capacity assignment in the population and  $CSA^k$  the cost of  $SA^k$ , where  $1 \leq k \leq Npop$ . Then,  $SA^k_{(lm)}$  is the spare capacity needed on the link between node  $l$  and node  $m$  in the  $k$ th spare capacity assignment. Figure 2 shows the basic Stage I algorithm.

### Stage I: Initial population generation

**Step 1** involves randomly choosing a link ( $jk$ ) in the backup topology matrix  $BTop$  then temporarily deleting it and checking if the topology of the virtual backup network is still 2-connected. The term 2-connected means that there are at least 2 disjoint paths between every node pair in the topology. If the backup topology matrix  $BTop$  is no longer 2-connected then the deleted link is returned and Step 1 is repeated.

**Step 2** enacts the successful repetition of Step 1, until the number of deleted links equals a specified input parameter *delete*.

Note, that Step 1 and 2 in effect randomly reduce the path set to be considered for routing backup connections, in contrast many IP formulations reduce the path set by imposing hop count limits.

**Step 3** determines the backup paths  $BP^k$  and spare capacity requirements  $SA^k$  for a specific failure scenario and restoration level requirement. Here we discuss the case of 100% restoration of all traffic affected by the failure of any single link in the network; other scenarios are given in [16] and require slight modifications. The  $BP^k$  and  $SA^k$  are found by applying the following steps, with  $f=1$  initially:

*Step 3-1.* Fail link  $f$  in the network by removing it from the topology matrix  $Top$  and determining which working paths in  $WP$  are affected by the failure. Also remove the failed link from the current backup topology matrix  $BTop$ .

*Step 3-2.* For a source destination pair  $ij$  affected by the failure, remove the working path  $WP_{ij}$  between node  $i$  and node  $j$ , by deleting all the links of  $WP_{ij}$  from the backup topology matrix  $BTop$ .

*Step 3-3.* Find backup path  $BP_{ij}$  between node  $i$  and node  $j$  using the current backup topology matrix  $BTop$  to determine the possible path set and shortest path routing to find the actual path. Determine the spare capacity requirements for the links on the backup path.

*Step 3-4.* Return the links of the working path  $WP_{ij}$  between node  $i$  and node  $j$  to the current backup topology matrix  $BTop$ .

*Step 3-5.* Repeat Steps 3-2 to 3-4 for every source destination pair affected by the failure of link  $f$ , and adjust the spare capacity requirements for each link. That is, if a link ( $lm$ ) is used on multiple backup paths for the same link failure case then the spare capacity requirement at that link is the sum of the spare capacities needed by each backup path passing through the link.

*Step 3-6.* Repeat Steps 3-1 to 3-5 for every link in the network  $Top$  (i.e., increment  $f$ ) and adjust the spare capacity requirements. The final spare capacity requirement on a link ( $lm$ ) is the maximum spare capacity required by the backup paths using that link for any single link failure  $f$ .

The steps above result in each backup path  $BP_{ij}$  between a pair of nodes  $i$  and  $j$  being link disjoint with its corresponding working path  $WP_{ij}$  thus allowing implementation of path restoration with link disjoint routes.

**Step 4** calculates the cost  $CSA^k$  of the current spare capacity assignment  $SA^k$ .

**Step 5** returns the links deleted in Step 1 to  $BTop$  resulting in  $BTop = Top$ .

**Step 6** repeats Steps 1-5, until the number of spare capacity assignments generated equals the specified population size  $Npop$  - an input parameter.

## Stage II Genetic Search

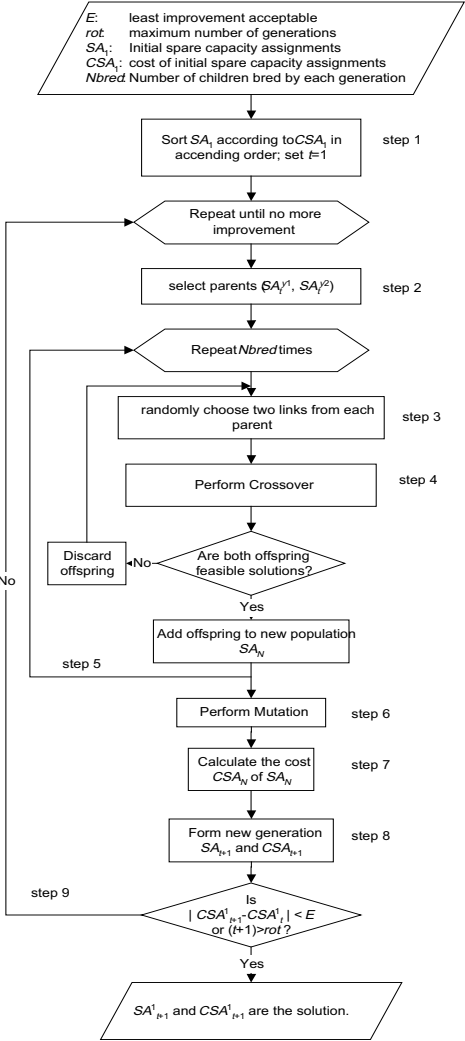
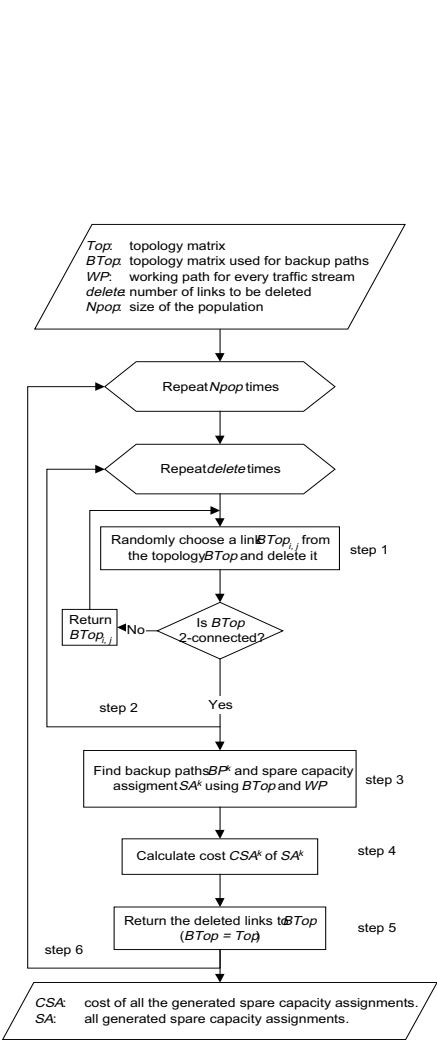
Stage II performs a GA search on the population generated from Stage I, until a stopping criterion is met. Fig. 3 shows the basic algorithm.

**Step 1** sorts the  $Npop$  spare capacity assignments  $SA$  in ascending order according to their costs  $CSA$  to form the initial generation  $SA_t$  with cost  $CSA_t$  for  $t=1$ .

**Step 2** selects pairs of the  $Npop$  spare capacity assignments for breeding. The pairs  $(SA_t^{y1}, SA_t^{y2})$  were selected according to a quadratic procedure by applying the following equation twice to get  $y1$  and  $y2$ .

$$y = \lceil x^2 \rceil \text{ where } x \text{ is a real number uniformly distributed between } [0, \sqrt{Npop}]$$





This approach is biased towards choosing the better solutions in the current generation for breeding. Steps 3 and 4 perform breeding and are called the crossover steps in GA terminology. These steps combine elements of two existing spare capacity assignments (parents) to produce two new spare capacity assignments (offspring) that are added to the population for possible continuation into the next generation. Several types of crossover operators were tested to find the operator best suited for our problem domain [16]. The crossover operator that was selected was two-position random crossover as it produced the highest rate of offspring which were feasible solutions.

**Step 3** randomly selects two links from each parent, that is links ( $jk$  and  $lm$ ) for  $SA_i^{y1}$  and links ( $pq$  and  $rs$ ) for  $SA_i^{y2}$ .

**Step 4** performs two-position crossover by switching the spare capacity assignment on the chosen links of the parents to create a pair of offspring ( $SA_i^{y1o}$ ,  $SA_i^{y2o}$ ). The child  $SA_i^{y1o}$  is equivalent to the parent  $SA_i^{y1}$  except the spare capacity value for link  $jk$  is taken from link  $pq$  of the other parent (i.e.,  $SA_i^{y2}$ ). Similarly, child  $SA_i^{y2o}$  is equivalent to the parent  $SA_i^{y2}$  except the spare capacity value for link  $rs$  is taken from link  $lm$  of the other parent (i.e.,  $SA_i^{y1}$ ). If either of the two offspring has a spare capacity assignment which is not a feasible solution for the required restoration level, then both offspring are discarded and Steps 3 and 4 are repeated. Otherwise the two offspring are added to the population of new solutions  $SA_N$ .

**Step 5** repeats Steps 2 to Step 4  $Nbred$  times, where  $Nbred$  is a input parameter determining the number of children bred by each generation.

**Step 6** performs mutation by first selecting a specific percentage of the current population, set by  $mrate$  (the mutation rate in GA terminology). Then the spare capacity values for the selected solutions are altered randomly with mutation probability  $mprob$ . The resulting alternated solutions, if feasible, are added to the population of new solutions  $SA_N$ .

**Step 7** calculates the cost  $CSA_N$  for all the new spare capacity assignments  $SA_N$ .

**Step 8** forms a new generation of spare capacity assignments  $SA_{t+1}$ . First, by copying the  $Nreprod$  best assignments from the current generation to the new generation along with their cost. Note  $Nreprod$  is a user input parameter. Second, the new solutions from  $SA_N$  are added to the new generation along with their cost  $CSA_N$ . The new generation is then sorted in ascending order according to the cost and the first  $Npop$  spare capacity assignments are retained in the next generation with the rest being discarded.

**Step 9** repeats Steps 2-8, until the improvement in the spare capacity cost between successive generations is less than  $E$  or the number of the generations created equals  $rot$ . Both  $E$  and  $rot$  are input parameters.

### 3 Numerical Results

Extensive numerical experimentation was conducted with the GA approach presented above for a variety of network topologies, loading conditions, failure scenarios and restoration requirements with details in [18]. The experimentation had two purposes, namely: (1) selection of parameter values and sensitivity analysis for the GA method and (2) comparison with existing spare capacity planning methods.

In the first set of experiments, the effects of different population sizes, number of links deleted in determining the initial population, reproduction rates, crossover algorithms, mutation rates, mutation probabilities and stopping criteria were studied. The parameters governing the performance of the Stage I algorithm outlined above are the population size,  $Npop$ , and  $delete$  the number of links deleted randomly from the virtual backup topology before determining the backup paths. Consider a network topology with  $N$  nodes and  $L$  links. We assume that the network nodes are at least two connected so the average network node degree  $deg$  ranges from 2 for a ring to  $(N-$

1) for a full mesh. The corresponding number of links  $L$  ranges from  $N \leq L \leq (N(N-1)/2)$ . Obviously, as the number of links and the average nodal degree increases, the greater the spare capacity assignment search space and therefore  $Npop$  and  $delete$  should increase. Numerical experimentation showed that a population size in the range  $Npop \in [(\lceil deg \rceil - 1)L, \lceil deg \rceil L]$  worked well. For the number of links to delete, it was found that the formula  $delete = L - (N-1) - \lceil (L-N)/(deg-1) \rceil$  yielded good results.

In considering the computational complexity of Stage I, we note that Steps 1 and 2 are repeated a random number of times until a feasible backup topology is found with  $delete$  links removed. The worst case number of repetitions is  $L$ . The most complex part of Step 1 is the test for a 2-connected network which is  $O(L)$  [19] complexity. Thus, Steps 1 and 2 are  $O(L^2)$ . Step 3 involves finding the backup paths and spare capacity assignments. This involves the repeated application of a shortest path algorithm which has complexity  $O(L+N \times \log(N))$  for all  $(N \times (N-1))$  source destination pairs and all  $L$  possible link failures. Thus, Step 3 is  $O((L \times N^2 \times (L+N \times \log(N))))$ . Steps 4 and 5 involve the calculation of the spare capacity cost for the link assignments and reinitializing the backup topology, the complexity is  $O(L)$ . Step 6 repeats Steps 1-5  $Npop$  times. Thus for Stage I, the complexity is  $O(Npop \times L \times N^2 \times (L+N \times \log(N)))$ .

For Stage II, the input parameters are  $Nbred$ ,  $Nreprod$ ,  $mrte$ ,  $mprob$ ,  $E$  and  $rot$ .  $Nbred$  determines the number of children created for consideration in each new generation and was selected to keep the population size constant ( $Nbred = (Npop - Nreprod) / 2$ ). Since the GA above uses an elitist reproduction approach the value of  $Nreprod$  was set to a constant value in each experiment. The numerical results showed that the quality of the solution was best when  $Nreprod$  was in the range  $Nreprod \in [10\%, 50\%]$ . The values for the mutation rate,  $mrte$ , and the mutation probability,  $mprob$ , were varied over a wide range ( $mrte \in [0.05\% - 10\%]$ ,  $mprob \in [0.005, 0.05]$ ) with no significant effects on the quality of the GA. This may be due to the random nature of the crossover algorithm used or the small number of generations considered. Since the GA was relatively insensitive to the choice of mutation parameters, values of  $mrte = 1\%$ ,  $mprob = 0.05$  were used. Note,  $E$  and  $rot$  specify the stopping criteria of the GA search and are largely determined by computer runtime and optimality considerations. One can see that increasing  $E$  and decreasing  $rot$  results in longer computer runtimes with possibly better final results. Here,  $E = 1 \times 10^{-5}$  and  $rot = 10$ , which results in the stopping criteria being  $rot$  for the results shown.

Considering the complexity of Stage II, Step 1 involves a simple sort of  $Npop$  values, for which the quick sort algorithm has  $O(Npop \times \log(Npop))$ . In Step 2, pairs of the existing population are randomly selected for breeding which is  $O(1)$ . The most computationally intensive steps in Stage II are Steps 3 and 4. The pairs of links in the parents are randomly selected and switched to create offspring. The worst case for this algorithm is  $O(L^2)$ . The two offspring are then tested for feasibility. This involves considering all the failure scenarios and checking whether the required restoration level is met by examining the spare capacity assignment on backup paths for all the source destination pairs. Here, we are considering 100% restoration for any single link failure, so the test requires considering every link and in the worst case every source destination pair for each link failure. The resulting Step 3 and 4 algorithms have  $O(L^2 \times L \times N^2 \times (L+N \times \log(N)))$  complexity. Note that Step 5 invokes the

repetition of Steps 3 and 4 until *Nbred* feasible solutions are found. In general the number of solutions that need to be created and tested to result in *Nbred* feasible solution is a random quantity with worst case *Npop*. Thus Steps 3-5 have complexity  $O(Npop \times L^3 \times N^2 \times (L + N \times \log(N)))$ . Step 6 involves performing mutation and a test for feasibility of the mutated solution resulting in an  $O(Npop \times L \times N^2 \times (L + N \times \log(N)))$  complexity. Steps 7 and 8 consist of calculating the cost of the various spare capacity assignments and forming a new generation and the complexity is  $O(L \times Npop)$ . Step 9 involves testing if a stopping criterion is met. If we assume the stopping criterion is the maximum number of generations *rot*, then the overall complexity of Stage II is  $O(rot \times Npop \times (\log(Npop) + L^3 \times N^2 \times (L + N \times \log(N))))$ . Note that both stages of the genetic algorithm method have a polynomial time complexity.

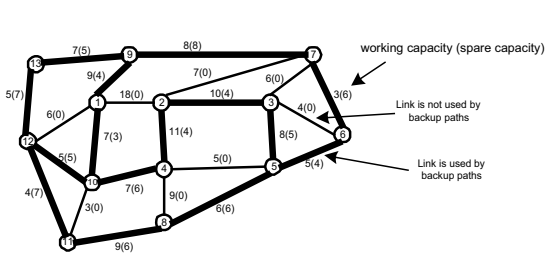


Fig. 4. Network 1 (13 nodes, 23 links)

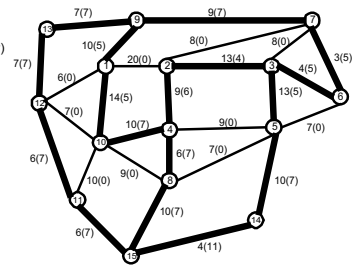


Fig. 5. Network 2 (15 nodes, 27 links)

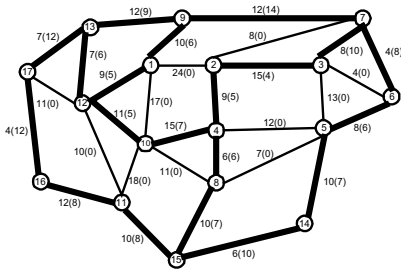


Fig. 6. Network 3 (17 nodes, 31 links)

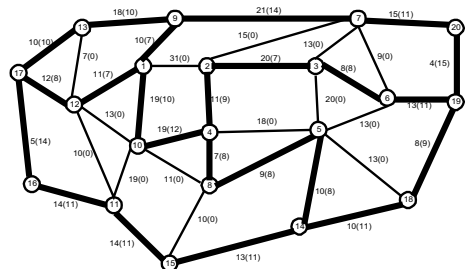


Fig. 7. Network 4 (20 nodes, 37 links)

Table 1 shows a few typical GA spare capacity results for the four networks with topologies shown in Fig. 4 – 7. One traffic demand (e.g. STS-1) between each node pair in the network was assumed. The allocated working capacity for each network was determined using shortest path routing. The working capacity and the spare capacity required to provide fault tolerance for any single link failure are marked beside each link as the first and second number respectively in Fig. 4–7. The bold lines indicate links with non-zero spare capacity. Notice that the spare capacity is concentrated on a subset of the links in the network with some links having zero spare capacity. In determining the results shown in Fig. 4-7, the GA tried to minimize the total network cost when a nonlinear capacity cost function was employed. The total link cost is the cost of the total capacity as determined by the summation of both the working and the spare capacity in the link. The parameters for the piece-wise cost

function are selected from an actual network service provider charges for bandwidth namely: 3.7, 5.8, 19.6, 64.2 for link units of OC-1, OC-3, OC-12, OC-48 respectively [17]. In order to permit a fair comparison with other approaches, the path set used for backup topologies in the GA was hop count limited with a limit of 7.

**Table 1.** Summary of the four networks studied

Network	Number of nodes	Number of links	Average node degree	Number of source-destination pairs	Total network load
1	13	23	3.54	78	156
2	15	27	3.60	105	210
3	17	31	3.65	136	272
4	20	37	3.70	190	380

In the second set of experiments the GA approach was compared with three methods from the literature: (1) the Spare Link Placement Algorithm (SLPA) heuristic [5]; (2) Link restoration using Integer Programming (Link IP) [2]; and (3) Path restoration with link disjoint routes using Integer Programming (Path IP) [3, 5]. The Spare Link Placement Algorithm (SLPA) is a heuristic approach with polynomial time complexity, and has been used by several telecom network operators for spare capacity planning. The Link IP approach reroutes all the affected traffic demands locally around the failure. As noted earlier, the Path IP approach is expected to provide the best results since it reroutes failed traffic over the remaining paths between each source destination pair after a failure. Both the Link IP and the Path IP techniques were solved using the commercial optimization package CPLEX. Our implementations of the three methods above were validated by reproducing published results from the literature. Note that a lower bound on the spare capacity requirements can be found by taking the Path IP formulation and relaxing the integer requirement, i.e., solving a Linear Programming version of the problem.

**Table 2.** Total spare capacity

Net- work	Working Capacity	Spare Capacity Bound	Spare Capacity, % Redundancy							
			SLPA		Link IP		GA		Path IP	
1	324	129.7	230	70.99	199	61.42	158	48.77	135	41.67
2	464	175.0	322	69.40	264	56.90	218	46.98	176	37.93
3	640	236.0	444	69.38	353	55.16	310	48.44	236	36.88
4	966	349.8	684	70.81	535	55.38	460	47.62	350	36.23

In Table 2, the total spare network capacity for 100% restoration for any single link failure as determined by the four algorithms using a hop count limit at 7 is given. The lower bound on the spare capacity requirements is found by solving the LP relaxation of the Path IP model is also shown. The % redundancy as measured by the ratio of spare capacity to the working capacity is also given in Table 2. For the Link IP and Path IP approaches the objective is to minimize the total spare capacity in the network. Compared to the Path IP results, the genetic algorithm has about 7–12% higher redundancy in the four networks. However, it results in 7–23% less redundancy than the spare capacity assignments given by the Link IP and SLPA approaches. Note, that the GA approach was designed to minimize the cost not the amount of spare capacity.

Neither the Link IP nor Path IP methods can deal directly with a nonlinear cost function. Adapting IP algorithms to nonlinear cost functions usually leads to an explosion of the search space size. On the contrary, whether the cost function is linear or not makes no difference to the GA approach. In order to compare the Link IP, Path IP and SLPA approaches cost, we solve the problem in two steps. The first step utilizes the algorithms to minimize the spare capacity. The second step minimizes the total network cost by fixing the link capacity at the level found in the first step and then finding the combination of capacity assignments that minimizes the cost. For example if a link requires 9 units of capacity this could be achieved by either 9 OC-1 or 3 OC-3 links, here the minimum cost option of 3 OC-3 links would be selected. Table 3 shows the total network costs from the above algorithms. The least cost solution is obtained by the Path IP method. The Link IP and SLPA yield solutions with a higher cost than the GA method or the Path IP method. Notice, that the GA approach can produce a cost close to that given by the Path IP method (within 3-7%).

**Table 3.** Total Spare Capacity Cost

Network	Total cost			
	SLPA	Link IP	GA	Path IP
1	1033	970	883	859
2	1467	1387	1262	1183
3	2024	1829	1698	1591
4	2960	2687	2516	2347

**Table 4.** Execution times

Network	Execution Time (minutes)				
	LP bound	SLPA	Link IP	GA	Path IP
1	0.2	1.02	1.0	0.48	91.7
2	0.5	4.33	3.01	0.65	367
3	2.5	7.07	8.17	0.73	833
4	14.67	16.67	20.17	1.38	5833

The primary benefit of the GA approach is its computational efficiency and scalability. This can be seen by examining the computer run times of the four methods for the networks studied as shown in Table 4. Both the SLPA and GA methods were implemented in PASCAL and executed on a Pentium II 400 MHz PC with 128 MB RAM running Windows NT 4.0. The Link IP and Path IP methods and LP relaxation of the Path IP model were implemented using the commercial package CPLEX 6.0 running on a SUN Enterprise 4000 server (10 parallel UltraSPARC-II 250 MHz CPUs with 2.6GB RAM). Thus the execution times shown below are not directly comparable as they are on different platforms but the scalability of each scheme can be inferred. From the results one can see that the execution time of the GA approach is the least affected by increases in the network size. Additional, numerical results for other network topologies including networks of more than 100 nodes (with average node degree 3.2) are given in [18].

## 4. Conclusions

In this paper, we proposed a genetic algorithm for spare capacity planning in STM networks based on link disjoint path restoration. The primary advantages of the GA method were its scalability and the capability of incorporating nonlinear cost functions. Parameter selection and the computational complexity of the GA approach were discussed. Numerical results illustrating the GA method solution were presented and compared to existing approaches from the literature. It was shown that the GA method provides a reasonably good spare capacity assignment with substantial computational savings.

## References

1. Doverspike, R., Wilson, B.: Comparison of Capacity Efficiency of DCS Network Restoration Routing Techniques. *Journal of Network and System Management*. Vol. 2. No. 2. (1994) 88-99
2. Herzberg, M., Bye, S., Utano, U.: The Hop-Limit Approach for Spare-Capacity Assignment in Survivable Networks. *IEEE/ACM Transactions on Networking*. December (1995) 775-784
3. Iraschko, R., MacGregor, M., Grover, W.: Optimal Capacity Placement for Path Restoration in STM or ATM Mesh Survivable Networks. *IEEE/ACM Transactions on Networking*. June (1998) 325-336
4. Herzberg, M., Wells, D., Herschtal, A.: Optimal Resource Allocation for Path Restoration in Mesh-Type Self-Healing Networks. *Proceedings of 15<sup>th</sup> International Teletraffic Congress*. Washington, D.C. June (1997)
5. Grover, W., Iraschko, R., Zheng, Y.: Comparative Methods and Issues in Design of Mesh-Restorable STM and ATM Networks. *Telecommunication Network Planning*. Kluwer Academic Publishers (1998)
6. Xiong, Y., Mason, L.: Restoration Strategies and Spare Capacity Requirements in Self-Healing ATM Networks. *IEEE/ACM Transactions on Networking*. Vol. 7, No. 1. Feb. (1999)
7. Sakauchi, H., Nishimura, Y., Hasegawa, S.: A Self-Healing Network with an Economical Spare-Channel Assignment. *Proceedings IEEE Global Conf. on Communications*. November (1990) 438-442
8. Venables, B., Grover, W., MacGregor, M.: Two Strategies for Spare Capacity Placement in Mesh Restorable Networks. *IEEE Global Conference on Communications*. November (1993) 267-271
9. Yamada, J.: A Spare Capacity Design Method for Restorable Networks. *Proceedings IEEE Global Conference on Communications*. November (1995) 931-935
10. Caenegem, B., Wauters, N., Demeester, P.: Spare Capacity Assignment for Different Restoration Strategies in Mesh Survivable Networks. *Proceeding IEEE International Conference on Communications*. June (1997) 288-292
11. Caenegem, B. V., Parys, W. V., Turck, F. D., Demeester, P. M.: Dimensioning of Survivable WDM Networks. *IEEE Journal on Selected Areas in Communications*. Vol 16, No. 7. (1998)
12. Norman, B., Bean, J.: Random Keys Genetic Algorithm for Complex Scheduling Problems. *Naval Research Logistics*. Vol. 46. (1999) 199-211
13. Hadj-Alouane, B., Bean, J., A Genetic Algorithm for the Multiple Choice Integer Program. *Operations Research*. Jan.-Feb. (1997) 92-101
14. Ko, K.-T., Tang, K.-S., Chan, C.-Y., Man, K.-F., Kwong, S.: Using Genetic Algorithms to Design Mesh Networks. *IEEE Computer*. August (1997) 56-60
15. Medhi, D., Tipper, D., Some Approaches to Solving a Multi-Hour Broadband Network Capacity Design Problem. to appear *Telecommunication Systems*
16. Goldberg, D.: *Genetic Algorithms in Search, Optimization and Machine Learning*. Addison-Wesley Publishing (1989)
17. McDysan, E., Spahn, D. L.: *Hands-on ATM*. McGraw-Hill (1998)
18. Al-Rumaih, A.: A Spare Capacity Planning Methodology for Wide Area Survivable Networks. Ph.D. Dissertation. Department of Information Science and Telecommunications. University of Pittsburgh. May (1999)
19. Kershenbaum, A.: *Telecommunication Network Design Algorithms*. McGraw-Hill (1993)

# A Cooperative Game Theory Approach to Resource Allocation in Wireless ATM Networks

Xinjie Chang<sup>1</sup> and Krishnappa R. Subramanian

Network Technology Research Center, School of EEE,  
Nanyang Technological University, Singapore 639798  
changxj@ieee.org  
esubram@ntu.edu.sg

**Abstract.** In the emerging wireless ATM networks, resource allocation with handoff consideration plays an important role in the quality of service (QoS) guarantee for multiple traffic sources. As efficiency is an important performance issue which is most widely used, the concept of fairness should also be considered. In this paper we investigate a fair and efficient resource allocation scheme for two types of traffics contending for the shared network resource. Based on the cooperative game theory, we model the fair and efficient allocation problem as a typical bargaining and arbitration problem, while the issues of efficiency and fairness are considered simultaneously by using the axiom approach. By modeling the corresponding queuing system as a Markov chain and using the Markov decision process (MDP) analysis, we convert the solution of optimal allocation policies into a typical linear programming problem for which the well-known simplex type algorithms can be easily implemented. Simulation results are also provided.

## 1 Introduction

Asynchronous Transport Mode (ATM) networks and wireless (cellular) communications networks are merging to form wireless ATM networks. While this integration can provide many benefits, it also leads to many technical challenges in implementation. In the interconnected heterogeneous networks the quality of service (QoS) issue needs to be resolved for different classes of traffic, as the performance criteria differ widely from low speed, less reliable cellular systems, to high speed ATM networks. Since wireless communication systems are bandwidth (network resource) limited, it is important from the network management standpoint to maximize the utility of network resources. In order to provide QoS guarantee for multiple traffic classes in a wireless ATM network, resource allocation is one of the central issues to be studied.

Resource allocation has been extensively studied [2]. Unfortunately, most of the schemes are designed for ATM networks and with the assumption that the terminals are fixed. As smaller cells are always used in wireless communication systems to

---

<sup>1</sup> Xijie Chang is currently working in the Information & Telecommunication Technology Center (ITTC), University of Kansas and can be reached at: changxj@itcc.ukans.edu.



meet the need for increased system capacity and spectrum efficiency, the number of cell boundary crossing increases significantly and thus the need for frequent and efficient handoff also increases. As a result, network resource allocations with handoff consideration in wireless ATM networks need to be resolved.

As there are many types of traffics with frequently conflicting performance criteria need to be supported on the limited network resources, multiple objective optimization techniques are needed to get the optimal allocation scheme [11]. Game theory provides a strong mathematical tool for this kind of problem. In early works, attention was mostly focused on the concept of efficiency, such as network resource utilization or throughput maximization. Based on the competitive game theory, the Nash equilibrium solution and Pareto-optimality were used to achieve efficiency. Although this solution can achieve global optimization, it is still shown that there are some cases in which the performance for one or some of the users can be very bad [10]. That is, the performance of some users may be sacrificed to achieve system efficiency, which is unfair to those users. As the quality of service required by various traffic types should be maintained, the concept of fairness is attracting more and more attention. The cooperative game theory provides a strong mathematical tool for the synthesis and analysis of the fair-efficient allocation schemes for the multi-service network [3]. In this paper, we present an optimal resource allocation scheme with handoff priority for wireless ATM networks based on this methodology. We consider two types of traffic to a network node: new calls and handoff calls with higher priority assigned to handoff traffics. By viewing the two types of calls as two players of a cooperative game, we model the problem as a two-person arbitration problem and use the Modified-Thomson scheme to solve optimal solution. Furthermore, by considering the corresponding queuing network as a Markov chain and using the Markov decision process (MDP) analysis, we convert the problem considered into a typical linear programming problem and use the well-known simplex-type algorithm to get the optimal allocation policies for each state. In our analysis, both the efficiency and the fairness features are easily considered by the axiom approach.

This paper is organized as follows: Section 2 provides a brief review of the cooperative game theory, especially the bargaining models and the arbitration schemes. In Section 3, the network model is described and the Modified-Thomson scheme based on the MDP analysis is presented. As an example, some numerical results are provided in Section 4, which illustrate the advantages of our method. Finally, the discussion and the directions for the future work are presented in Section 5.

## 2 Cooperative Game Theory: Bargaining and Arbitration Model

In the cooperative game, players are assumed to be free to cooperate and bargain to obtain mutual advantage in contrast to the non-cooperative (competitive) game case, in which context communications are not allowed [4]. As bargaining and arbitration are most often used in this kind of problem, we first introduce these two concepts in the following subsections.

## 2.1 Bargaining Problem

Cooperative games [5] concern at least two players, who may try to reach agreement that can give them mutual advantages. In two-person cooperative game, we can use a region  $R$  to denote the problem set and use  $(u, v)$  to denote a point in  $R$  while  $u$  and  $v$  represent the two players' utility, respectively. In bargaining situations, the players will act cooperatively to discard all jointly dominated pairs (a pair  $(u, v)$  is said to be dominated by another pair  $(r, s)$  if both  $u \leq r, v \leq s$ ) and all undominated pairs which fail to give each of them at least the amount he could be sure of without cooperation.

It is well known that if the players restrict their attention to the bargaining set and bitterly bargain over which point to select, then rational players will frequently fail to reach an agreement [5]. For this case, we can assume that the players will be willing to resolve their conflict through an arbiter—an impartial outsider who will sincerely envisage his mission to be “fair” to both players. Thus, we can use the so-called arbitration schemes to get the optimal allocation solution.

## 2.2 Arbitration Scheme

One of the favorite approaches to get the arbitration solution is the so-called axiomatic approach that exams the subjective intuition of “fairness” and optimality, and formulates these as a set of precise desiderates that any acceptable scheme must fulfill. Once these axioms are formalized, then the original problem is reduced to a mathematical investigation of the existence and characterization of arbitration schemes.

Nash studied this kind of problem in 1950[5] and provided a formal definition and solution. The bargaining problem  $[R, (u^0, v^0)]$  is characterized by a region  $R$  of the plane and a starting point  $(u^0, v^0)$ , and we can use  $F$  to denote an arbitration scheme which maps a typical bargaining game into an arbitrated outcome. Four axioms are defined:

**Invariance (inv):** The arbitrated value is independent of the unit and origin of the utility measurement (invariance with respect to linear transform).

**Independence of irrelevant alternatives (iaa)** If some of the possible utility combinations were dropped from  $R$ , resulting a subset  $R'$ , then if  $(u^*, v^*)$  still left in the  $R'$ , then arbitration value for the new problem will not change.

**Efficiency--Pareto-optimality (po):** No one can be made better without making the other worse off.

**Symmetry:** If  $R$  is symmetry about the axis  $u=v$  and  $(u^0, v^0)$  is on the axis, then the solution point is also on the axis.

These four axioms fully characterized the Nash solution. Besides Nash's scheme, some alternative solutions schemes have also been reported [5]. By introducing the concept of preference functions, Cao[6] also formed a mathematical interpretation of the arbitration schemes.

Definition 1: preference function

Player  $i$ 's preference function  $V_i$  for the outcomes of a bargaining problem  $[R, (0,0)]$  is a function of the utilities  $u$  and  $v$  and the set  $R$ , i.e.,  $v_i = f_i(u, v, R)$ , where  $(u, v) \in R$ .

**Note:** under the axiom (*inv*), without losing any generality, we can always linearly transform any pair  $[R, (u^0, v^0)]$  in such a way that:

$$(\max(u' : u' \in R) = 1, \max(v' : v' \in R) = 1) \& (u^0, v^0)' = (0, 0) \quad (1)$$

Which is called normalization.

For our problem, the preference function can be defined as [3,8]:

$$v_1 = u + \gamma(1 - v) \quad (2)$$

$$v_2 = v + \gamma(1 - u) \quad (3)$$

where  $\gamma$  is a weighting factor and  $\gamma = -1, 0, 1$  will correspond to the Modified-Thomson, Nash and Raiffa schemes, respectively. The optimal solution is to maximize the product of player's preference functions:

$$(u^*, v^*) = \arg(\max_{(u, v)} \{v_1 \cdot v_2\}) \quad (4)$$

In the Modified-Thomson case, as  $\gamma = -1$ , then  $v_1 = v_2$ , therefore the problem can be reduced to the maximization of  $u+v$ , which will make the problem much simpler to solve.

### 3 Network Model

In mobile communications systems, there are generally two categories of arrivals to a link (i.e., normal and handoff arrivals). In most cases preference should be given to handoffs over normal arrivals (handoff priority schemes), because the dropping of an existing connection will have more impact on performance than the blocking of a new call. The simplest scheme is fixed channel reservation (i.e., to assign  $C_h$  channels exclusively for handoff arrivals among the total  $C$  channels in a cell) [7]. Being a deterministic priority scheme, it is not efficient, so here we use another scheme described below.

To begin, we focus our attention on only one cell which can carry two types of arrivals: normal arrivals and handoff arrivals. We assume:

Normal arrivals offered to the network can be characterized by a Poisson process with average rate  $\lambda_n$ . The holding time can be modeled by an exponential distribution with mean  $\mu_n^{-1}$ . Bandwidth requirement for normal arrivals is denoted by  $d_n$ .

The handoff arrival is also characterized as a Poisson process with average rate  $\lambda_h$ , exponentially distributed holding time with mean  $\mu_h^{-1}$  and bandwidth requirement  $d_h$ . We assume here that:  $d_h > d_n$ .

The total number of channels that available at the cell is denoted by  $N$ . At any given instant  $t$ , let  $X(t) = \{x_n = i, x_h = j\}$  to denote the state of network, where  $x_n$  and

$x_n$  are the number of normal arrivals and handoff arrivals current in progress in the cell, respectively.

Therefore, the system state space can be denoted by

$$E = \{(i, j) : 0 \leq i, j, 0 \leq i * d_n + j * d_h \leq N\}$$

and can be divided into several subspaces [7]:

$$E = E_i + E_b + E_e \quad (5)$$

$$E_i = \{(i, j) : 0 \leq i, j, 0 \leq i * d_n + j * d_h \leq N - d_h\} \quad (6)$$

$$E_b = \{(i, j) : 0 \leq i, j, N - d_h + 1 \leq i * d_n + j * d_h \leq N - 1\} \quad (7)$$

$$E_e = \{(i, j) : 0 \leq i, j, i * d_n + j * d_h = N\} \quad (8)$$

If the state is in subspace  $E_e$ , then every coming arrival will be blocked due to lack of channel. In subspace  $E_b$ , only handoff arrival will be blocked due to no channel available and normal will be admitted with some probability. In subspace  $E_i$ , both kinds of arrival can be accepted with some probability. As handoff priority is considered, we assume that the handoff arrivals will always be admitted with probability 1, as long as there are enough channels available and only normal arrivals will be admitted with some probabilities  $\alpha(i, j)$ , which are the control parameters that need to be calculated.

Based on these assumptions, we can address the problem by using the Markov Decision Process (MDP) model [3,8]. As the state of network is represented by a vector  $x$  and the state space  $E$ . The access policy will be equivalent to an admission subset  $A \in E$ , and under this policy, an arrival is accepted when the state is  $x$  if and

only if:  $x + e_k \in A$ ,  $e_k = \begin{bmatrix} 0 & \dots & 1 & \dots & 0 \end{bmatrix}$ . We also can assume that the

departure is never blocked. Where the system is in state  $x$ , reject/accept decision must be made for each coming arrival. Thus, the action space becomes:

$$B = \{ (a_n, a_h) : a_n \in \{0,1\}, a_h \in \{0,1\} \} \quad (9)$$

The actual action space is a state-dependent subspace of  $B$ :

$$B_x = \{a \in B, a_n = a_h = 0, \text{ if } x + e_k \notin A\} \quad (10)$$

For a MDP model, we can rewrite the state as  $X_B = (x_n, x_h, x_\delta)$  with  $x_\delta$  is a random variable denoting the action of refusal or acceptance for connection of a normal arrival. We can also use a new state space expression:  $E_D = \{(x, a), x \in E, a \in B_x\}$ .

In order to get optimal scheme, we can use the average-reward MDP model, and a reward function  $r(x, a)$  should be chosen. We chose the reward function so that optimization objective is to maximize the product of the preference function of normal arrivals and handoff arrivals. Here, we first choose the Modified-Thomson scheme. Then the problem can be described as a linear programming model:

Max:

$$Z_D = \frac{\lambda_n d_n}{\mu_n N} \sum_{x \in E} p(x, 1) + \frac{\lambda_h d_h}{\mu_h N} \sum_{x \in E_i} (p(x, 0) + p(x, 1)) \quad (11)$$

Subject to:

$$\sum_{(x,a) \in E_D} p(x,a) = 1 \quad (12)$$

$$\sum_{a \in B_x} q(x,a,x) p(x,a) = \sum_{(x',a') \in E_D} q(x',a',x) p(x',a') \quad (13)$$

$$p(x,a) \geq 0, \quad \text{for all } (x,a) \in E_D \quad (14)$$

Where:

$$q(x',a',x) = \begin{cases} \lambda_n \cdot 1\{a' = 1, x' \in E_i + E_b\} & x' = (x_n - 1, x_h) \\ \lambda_h \cdot 1\{x' \in E_i\} & x' = (x_n, x_h - 1) \\ x'_n \mu_n & x' = (x_n + 1, x_h) \\ x'_h \mu_h & x' = (x_n, x_h + 1) \\ 0 & \text{otherwise} \end{cases} \quad (15)$$

$$q(x,a,x) = \lambda_1 \cdot 1\{a = 1, x \in E_i + E_b\} + \lambda_2 \cdot 1\{x \in E_i\} + x_1 \mu_1 + x_2 \mu_2 \quad (16)$$

Also, the blocking probability constraints can be included as:

$$\sum_E p(x,0) \leq B_n \quad (17)$$

$$\sum_{E_b} (p(x,0) + p(x,1)) + \sum_{E_e} p(x,0) \leq B_h \quad (18)$$

Then, the two-phase simplex algorithm can be used to find the optimal solution. After getting  $p(x,d)$ , the admission probabilities can be calculated by:

$$\alpha(i,j) = p(i,j,1) / (p(i,j,0) + p(i,j,1)) \quad (19)$$

## 4 Numerical Example

We constructed a network layer model in the OPNET environment for our simulation scenario, as shown in Figure 1. The network node consists of two traffic source modules, a switch and a sink module. The two traffic source modules generate Poisson type traffics that represent new calls and handoff calls, respectively. The switch is viewed as a finite buffer queue with finite capacity. The sink node is only used for simulation purpose, which can destroy the received packets so that there are enough system resources (memory) for the simulation program. The detailed internal process layer model for the admission control module is shown in Figure 2. In analysis, we assume that the arrival rate of handoff is known *a priori*. But in fact, the parameter is implicit and must be determined from the process dynamics [6].

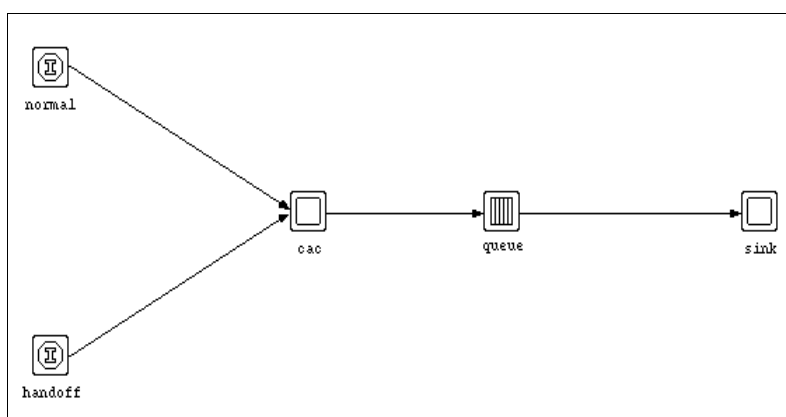
Here, the load is defined as:

$$u = d_n (\lambda_n / \mu) + d_h^* (\lambda_h / \mu) \quad (20)$$

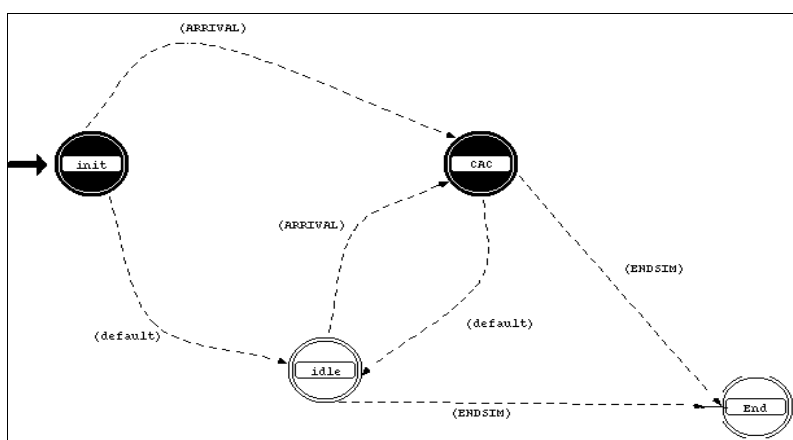
By increasing the arrival rate of these two sources (thus the load), we get the loss ratio increased as shown in Figure 3 and 4. Figure 3 gives an example of the variation of loss ratio  $b_1$  with the increasing of offered load where, for a value of offered load, an optimal control is computed. In the figure, the curve for the case without control is attached for comparison. Figure 4 shows the variation of loss ratio  $b_2$  against the changing of offered load, the curve for the case without control is also attached for comparison. In Figure 5, the improvement of system utilization against the offered

load is illustrated by the comparison of the cases without control and with optimal control. The following can be seen:

- 1) In the case without control, the handoff traffic always has a higher loss ration due to its higher bandwidth requirements.
- 2) The higher loss ratio of handoff traffic means unfair allocation.
- 3) In the case with optimal control, we can control the system so that both traffics have performance improvement and thus improvement on the system utilization.
- 4) Under our optimal control, the two types of traffic not only get performance improvement but also achieve similar performance, which is viewed as a fair allocation of resource.



**Fig. 1.** Node model



**Fig. 2.** CAC Process Model

5) We have to admit that although our scheme can achieve good performance under light and moderate traffic load, there do exist some cases, when the load is very high so that no feasible solution can be obtained. This is the one shortcoming of the scheme.

5 Conclusion

A fundamental problem in connection oriented multi-service networks is finding the optimal policy for call acceptance. In this article, we have presented a fair-efficient channel assignment policy based on the cooperative game theory, and especially, the arbitration scheme and axiom approach. Theoretical analysis and numerical results indicate that:

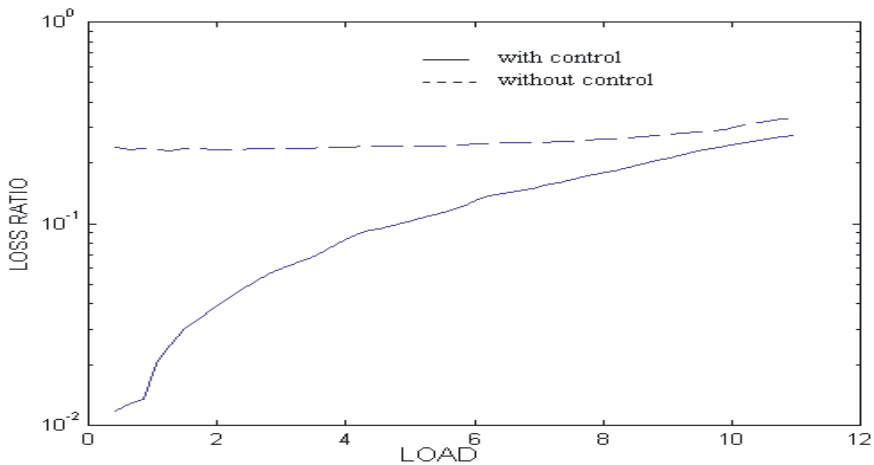


Fig. 3. Loss ratio against load (for handoff calls)

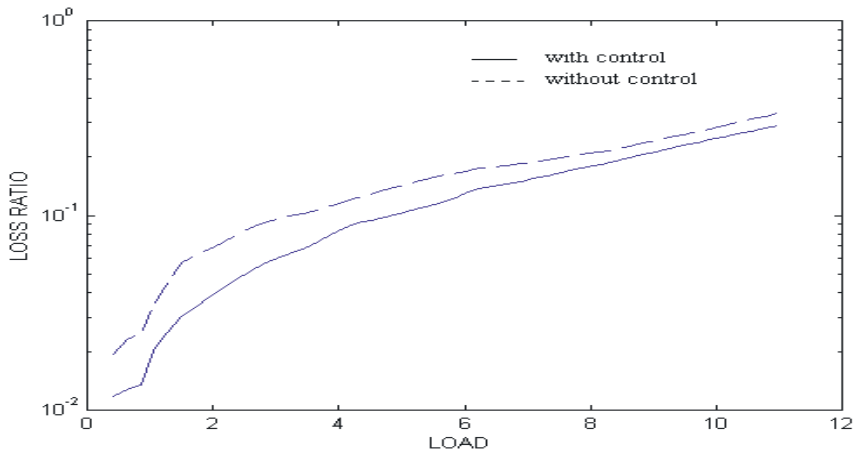
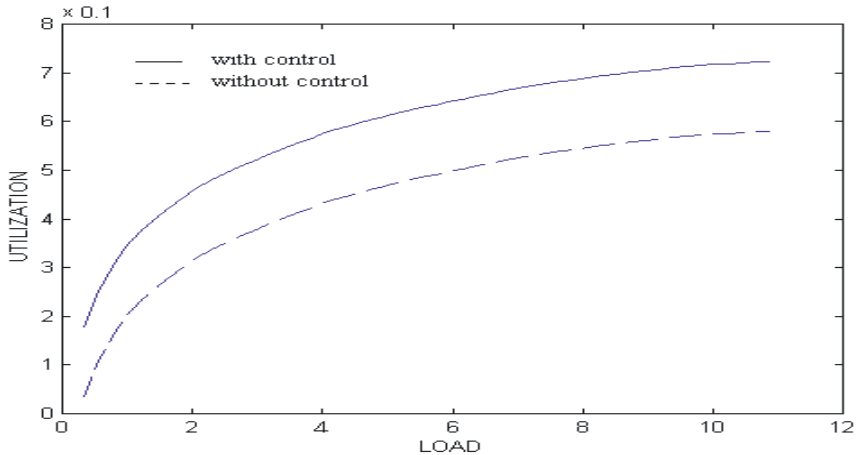


Fig. 4. Loss ratio against load (for new calls)



**Fig. 5.** Utilization improvement

An optimal resource allocation scheme is devised to maximize system utilization while maintaining a fair allocation of resources among competing users, satisfying their individual QoS requirements..

The arbitration scheme and axiom approach are quite appropriate and applicable to resolve these issues

Allocation policy for each state can be calculated by solving a typical linear programming problem while the well-known simplex type algorithms can be used.

After pre-calculation of allocation policies for each state, resource allocation scheme can be implemented on-line very easily and performed at high speed, only a simple table-search is needed.

While the model proposed in this paper can work well for some modest size networks (small state and admission policy spaces), the evaluation of optimal allocation for large-scale networks will become intractable due to the large state and policy spaces. In such cases, some state space reduction and approximation algorithms need to be investigated.

## References

1. G. P.Pollini, "Trends in Handover Design", IEEE Communications Magazine, Vol. 34, No.3, 1996
2. Xinjie, Chang, K.R.Subramanian, "Call Admission Control Policies for ATM Networks", WmATM'99, June, 1999, Silicon Valley, San Jose, USA, 1999
3. Z. Dziong, L. G. Mason, "Fair-Efficient Call Admission Control Policies for Broadband Networks—A Game Theoretic Framework", IEEE/ACM Transactions on Networking, Vol. 4, No.1, 1996
4. Alvin E. Roth, "Game-theoretic models of bargaining", Camb. [Cambridge], New York: Cambridge University Press, 1985.
5. Luce R. Duncan, Raiffa, Howard, "Games and Decisions: Introduction and Critical Survey", New York: Dover Publications, 1989, c1957.



6. X. Cao, "Preference Function and Bargaining Solution", Proceeding of IEEE CDC, 1982
7. Cong Li, "A Study on Dynamic Resource Allocation and Handoff in Cellular Radio System", M. Eng. Thesis, School of EEE, Nanyang Technological University, Aug, 1997
8. T. Oda, Yu Watanabe, "Optimal Trunk Reservation for a Group with Multi-slot Traffic Streams", IEEE Transaction On Communications", Vol.38, No.7, 1990
9. OPNET 3.5.A User Manual, MIL3 Company, 1997
10. D. Douligeris, R. Mazumdar, "More On Pareto-Optimal Flow Control", 26<sup>th</sup> Annual Allerton Conference on Communication, Control ad Computing, 1988
11. D. Douligeris, R. Mazumdar, "On Pareto-Optimal Flow Control in a Multiclass Environment", 25<sup>th</sup> Annual Allerton Conference on Communication, Control ad Computing, 1987

## Author Index

- Aalto, S., 275  
Abler, R., 458  
Ahn, G.-S., 311  
Aida, H., 74  
Akansu, A.N., 85  
Al-Raweshidy, H., 495  
Al-Rumaih, A., 957  
Altman, E., 567  
Alves, A.P., 895  
Andrade, M.T., 895  
Antoniou, Z., 470  
Asaka, T., 835  
Asano, S., 410  
Ast, L., 287  
Awduche, D.O., 144  
Ayad, A.S., 714
- Banerjee, G., 752  
Bao, T.T., 74  
Barakat, C., 567  
Barta, P., 727  
Baworntummarat, C., 189  
Beauquier, B., 859  
Becker, M., 423  
Berquia, A., 920  
Beylot, A.-L., 423, 776  
Biersack, E.W., 580  
Bíró, J., 727  
Björkman, M., 156  
Blondia, C., 264  
Boumerdassi, S., 776  
Boyer, J., 228  
Bruneel, H., 374, 433  
Brustoloni, J.C., 608
- Cain, B., 108  
Cali, F., 786  
Campbell, A.T., 311  
Candan, K.S., 859  
Canonico, R., 811  
Capone, J.M., 252  
Casals, O., 398
- Cerdán, F., 398  
Chahed, T., 240  
Chang, C.-J., 620  
Chang, J.-H., 702  
Chang, X., 969  
Chen, F., 85  
Choi, M., 458  
Cinkler, T., 287  
Clérot, F., 655  
Conti, M., 350, 786  
Copeland, J., 458  
Costa, L.H.M.K., 847  
Cselényi, I., 643
- Dagiuklas, T., 495  
Dallery, Y., 740  
Daniëls, T., 264  
De Vleeschauwer, D., 374  
Detken, K.-O., 883  
Dias de Amorim, M., 1  
Dinan, E., 144  
Domingo-Pascual, J., 23  
Douillet, P.L., 423  
Duarte, O.C.M.B., 1, 847  
Dugeon, O., 228
- El-Hadidi, M.T., 714  
Elsayed, K.M.F., 714  
Erramilli, A., 362  
Ethridge, J., 540
- Fayet, C., 240  
Fdida, S., 763, 847  
Fehér, G., 643  
Fei, A., 132  
Fenger, C., 168  
Feraud, R., 655  
Ferreira, A., 859  
Fiedler, M., 446  
Fulp, E.W., 945  
Furini, M., 483

- Gagnaire, M., 49  
 Garay, J.A., 608  
 Gerla, M., 132  
 Ghezzi, S., 350  
 Girard, A., 667  
 Gliese, U., 168  
 Gregori, E., 350, 786  
 Guillemin, F., 228  
 Gummalla A.C.V., 507  
 Gupta, P., 528
- Habib, I., 871  
 Hardjono, T., 120  
 Harju, J., 596  
 Hébuterne, G., 240, 740  
 Heikkinen, T., 679  
 Hoebeke, R., 49  
 Horvath, G., 410  
 Hu, X.-Y., 580  
 Hudson, I., 36
- Iamvasant, T., 189
- Jabbari, B., 144  
 Jaruvitayakovit, T., 299  
 Jeong, S.-H., 458  
 Jiang, S., 932  
 Johansson, N., 799
- Karsten, M., 325  
 Kermani, P., 871  
 Khemiri, M., 667  
 Kihl, M., 799  
 Körner, U., 799  
 Kone, M.T., 632  
 Konorski, J., 201  
 Krishnam, M.A., 252
- Labbé, C., 655  
 Lee, S.-B., 311  
 Li, B., 932  
 Liew, S.C., 11  
 Limal, E., 168  
 Limb, J.O., 507  
 Liu, Y., 957  
 Low, C.P., 213
- Luo, X., 932  
 Mahon, C.J., 168  
 Mangin, C., 228  
 Mangues-Bafalluy, J., 23  
 Mark, J.W., 338  
 Marosi, G., 410  
 Martin, S., 655  
 McKeown, N., 528  
 Melander, B., 156  
 Miklós, G., 689  
 Mikou, N., 920  
 Miyoshi, T., 835  
 Molnár, S., 689  
 Morino, H., 74
- Naghshineh, M., 871  
 Nakajima, T., 632  
 Nandy, B., 540  
 Narayan, O., 362  
 Neidhardt, A., 362  
 Németh, F., 727  
 Németh, K., 643  
 Nilsson, A.A., 620  
 Nitsos, K., 495  
 Norman, B.A., 957  
 Nyberg, E., 275
- Östring, S.A.M., 36  
 Onozato, Y., 908  
 Oueslati-Boulahia, S., 823  
 Owen, H., 458
- Pallou, D., 655  
 Pastoors, T., 555  
 Pavarangkoon, P., 299  
 Perennes, S., 859  
 Perros, H.G., 176  
 Petit, G.H., 374  
 Pieda, P., 540  
 Prapinmongkolkarn, P., 299  
 Pujolle, G., 1  
 Pulakka, K., 596
- Raatikainen, P., 98  
 Raatikainen, S., 98  
 Ramesh, S., 176  
 Rangsinoppamas, N., 299

Reeves, D.S., 945  
Ritter, H., 555  
Roberts, J.W., 823  
Rodriguez, P., 580  
Romano S.P., 811  
Rosenberg, C., 667  
Rosenberry, R., 752  
Rouskas, G.N., 176

Saito, H., 410  
Saito, T., 74  
Sa-niamsak, W., 299  
Saniee, I., 362  
Seddigh, N., 540  
Sellitto, M., 811  
Sen, A., 859  
Shen, X., 338  
Sherif, M., 871  
Sidhu, D., 752  
Simon, J.-L., 655  
Sirisena, H., 36  
Sokol, J., 458  
Stange, C., 518  
Stavrakakis, I., 470  
Steyaert, B., 374, 473  
Stojanovski, S., 49  
Subramanian, K.R., 969  
Suzuki, M., 386  
Szabó, R., 727  
Sze, H.-p., 11

Tanaka, Y., 835

Tassiulas, L., 702  
Tatai, P., 410  
Thng, I., 932  
Tipper, D., 957  
Towsley, D., 108, 483  
Tsuchiya, T., 410

Urvoy, G., 740  
Uzun, N., 85

Venkatachalam, A., 252  
Ventre, G., 811  
Viniotis, Y., 61  
Virtamo, J., 275  
Voos, H., 446

Walraevens, J., 433  
Wang, N., 213  
Wehrle, K., 555  
Wittevrongel, S., 374  
Wright, S., 61  
Wuttisittikulkij, L., 189

Yamamoto, U., 908  
Yousef, S., 518

Zeng, Y., 338  
Zhang, X., 311  
Zhou, J., 908  
Ziegler, T., 763